An Automated Scalable Framework for Distributing Radio Astronomy Processing Across Clusters and Clouds

Mechev, A.P.; Oonk, J.B.R.; Danezi, A.; Shimwell, T.W.; Schrijvers, C.; Intema, H.T.; ... ; Rottgering, H.J.A.

PROCEEDINGS OF SCIENCE

An Automated Scalable Framework for Distributing Radio Astronomy Processing Across Clusters and Clouds

A.P. Mechev*
Leiden University
E-mail: apmechev@strw.leidenuniv.nl

J.B.R. Oonk
Leiden University, ASTRON
E-mail: oonk@strw.leidenuniv.nl

A. Danezi
SURFsara
E-mail: anatoli.danezi@surfsara.nl

T.W. Shimwell
Leiden University
E-mail: shimwell@strw.leidenuniv.nl

C.Schrijvers
SURFsara
E-mail: coen.schrijvers@surfsara.nl

H.T. Intema
Leiden University
E-mail: intema@strw.leidenuniv.nl

A. Plaat
Leiden University
E-mail: a.plaat@liacs.leidenuniv.nl

H.J.A. Röttgering
Leiden University
E-mail: rottgering@strw.leidenuniv.nl

The Low Frequency Array (LOFAR) radio telescope is an international aperture synthesis radio telescope used to study the Universe at low frequencies. One of the goals of the LOFAR telescope is to conduct deep wide-field surveys. Here we will discuss a framework for the processing of the LOFAR Two Meter Sky Survey (LoTSS). This survey will produce close to 50 PB of data within five years. These data rates require processing at locations with high-speed access to the archived data.

To complete the LoTSS project, the processing software needs to be made portable and moved to clusters with a high bandwidth connection to the data archive. This work presents a framework that makes the LOFAR software portable, and is used to scale out LOFAR data reduction. Previous work was successful in pre-processing LOFAR data on a cluster of isolated nodes. This framework builds upon it and and is currently operational. It is designed to be portable, scalable, automated and general. This paper describes its design and high level operation and the initial results processing LoTSS data.

International Symposium on Grids and Clouds 2017 -ISGC 2017-
5-10 March 2017
Academia Sinica, Taipei, Taiwan

*Speaker.

1. Introduction

The LOFAR radio telescope is the world's largest aperture synthesis array with more than 20,000 antennas, and baselines of 60 m to 1000 km [1]. With its unprecedented sensitivity and angular resolution at ultra-low frequencies, LOFAR's goals are far reaching: from studying pulsars and supernova remnants in the Milky Way to the evolution of distant galaxies and the Epoch of Reionization [2]. Additionally, LOFAR is a pathfinder for the larger Square Kilometer Array (SKA) [3] radio telescope. At low frequencies, the SKA telescope is expected to increase the data rate [4] to 400TB per day creating more than 120PB per year [5].

The LOFAR Two Meter Sky Survey (LoTSS) [6] is observing 3000 different fields that will collectively map the entire northern radio sky. The majority of these datasets are anticipated to be 16TB per field. As such, the survey will create a total of 48 Petabytes. To complete the LoTSS survey in the project's target 5 year duration, $\sim$1PB of data must be processed each month. To mitigate delays caused by data transfer, processing must be done at a location with a high bandwidth connection to the raw data. SURFsara in Amsterdam is one such site and is also one of the LOFAR data archive locations.

Software packages for the initial processing of LOFAR data already exist [7], however they were not implemented to efficiently operate on all cluster architectures. Environments with isolated compute nodes are a case where the current processing cannot fully use the resources. To complete the LoTSS project, a framework is needed to automatically process multiple datasets at the SURFsara location. This location has a large computing capacity, and has previous success with LOFAR processing [8].

We've built a framework on top of the LOFAR DSP[1] platform [8] named the **LOFAR Reduction Tools** (LRT).The LRT software provides:

- Automation, enabling processing of multiple concurrent jobs
- Portability, enabling processing at different locations
- Scalability, enabling adding worker machines as required by the workload
- Generalization, enabling integration of software from other scientific domains

In this paper, we present the implementation of the LOFAR processing pipeline for Direction Independent calibration, also known as 'pre-FACTOR' [9], into this framework. This software has been in use since November 2016 and at the time of writing (March 2017) has processed more than 100 datasets. This corresponds to a rate of roughly one dataset per day. By deploying the LRT framework on a cluster with a high-bandwidth connection to the data, the entirety of the LoTSS data can thus be reduced within the five year timespan of the project.

The paper is structured as follows: Section 2 lists current work related to the research question. Section 3 outlines the LOFAR data reduction process and computational requirements. Section 4 describes the design of the LRT framework, its capabilities, the modification of the existing LOFAR software, performance and results. Finally, conclusions and future work are discussed in Section 5.

2. Related Work

Scientific projects have begun producing petabyte-size datasets [10]. With increasing data

[1] Distributed Shared Processing

sizes, researchers have begun focusing on scalable ways to parallelize their workflows. Because of this, processing is increasingly moving to grid- and cloud-based distributed computing facilities. From genetic sequencing [11] and bioinformatics [12] to neuroscience [13] and ecology [14], ever growing datasets have driven the development of distributed workflow systems in science [15] [16].

The framework presented in this publication is built on previous work distributing LOFAR pre-processing on a computing cluster [8] using a PiCaS server [17] to track progress. The details of this platform, the LOFAR DSP [8], are discussed elsewhere. Here we only provide a brief overview of the elements in this platform that interact with the LRT framework. The platform for LOFAR processing includes a PiCaS server and a CernVM Filesystem (CVMFS) client [18], previously deployed and tested on the target cluster. Additionally, continued technical support for this platform is provided by the SURFsara science support group.

PiCaS is a token pool database used to create tokens describing processing jobs. It is built upon the CouchDB database [19]. PiCaS [20] and CouchDB have been used in other distributed computing projects to launch and monitor jobs and store metadata. Job monitoring using PiCaS is also used in projects such as Sim-City [21] and Finite Element modelling for sea dyke design [22]. In these works, pilot jobs were automatically launched and tracked remotely using the PiCaS software.

CouchDB is also successfully used by the LHCb team to monitor the nightly software build process [23] and by Sante et al. [24] to launch asynchronous jobs to visualize and analyse gene sequencing data. As CouchDB documents can hold arbitrary information and attachments, CouchDB is suited for projects requiring the storing of metadata for many concurrent jobs, such as our application.

CVMFS [18] has been used by projects to package and publish software. The software used by many projects in High Energy physics, for example ATLAS [25] and the NOνA [26] groups. These groups compiled scientific software on a central server and publish it to worker nodes. The LOFAR software has been similarly packaged [8]. This makes deployment of processing scripts possible without a priori compilation on the distributed computing worker nodes.

3. LOFAR Data Processing

Creating images from LOFAR data requires several steps of calibration and imaging. In order to place this work in the proper radio astronomy context, a brief introduction to the data processing in the context of the LoTSS is presented below. Section 3.1 gives an overview of LOFAR processing from an archived observation to a final image (Fig.1). Section 3.2 details the processing steps currently implemented as well as their computational challenges. Section 3.3 contains an overview of the benefits of integrating the processing software with the LRT framework and a description of the processing by focusing on the dataflow (Fig. 2).

3.1 Producing Images From LOFAR Data

The raw LoTSS data is stored at two locations of the LOFAR Long Term Archive [27]. Typically each dataset is 16 TB and is split into 244 files of 65GB. Throughout the LoTSS data processing, this data is reduced to a $\sim$500GB set of calibrated data. The calibrated data is then imaged

producing a final set of several 1.2GB images of 25k×25k pixels each. The calibrated dataset is archived to allow for future refinement and re-imaging.

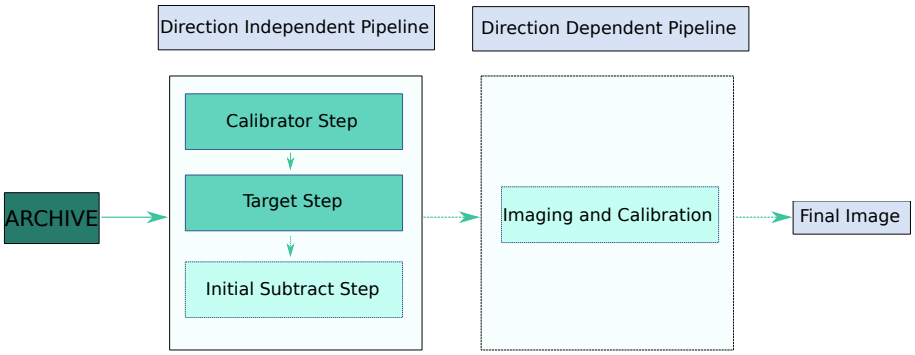


Figure 1: A schematic of the data flow through the Direction Independent Pipeline (pre-FACTOR [9]) and one of the Direction Dependent Pipelines, DDFacet (Tasse et al. in prep). The Initial Source Subtraction step is only necessary for some DD pipelines and is not currently implemented.

Data reduction for the LoTSS survey is split into two pipelines: Direction Independent (DI) pipeline followed by the Direction Dependent (DD) pipeline (Fig1). The Direction Independent pipeline [7] [9] is the first pipeline of LOFAR processing. It is necessary to produce a suitable starting point for the DD pipeline, however it produces images that are limited in resolution and contain residual instrumental effects [6] [7] (Fig7b). In the DD pipeline, to achieve high fidelity continuum images, the ionospheric and beam errors must be corrected [29]. The latter effects vary across the field of view. Here we present an implementation of the DI pipeline and the framework built to automate it. The LRT framework is built on top of the LOFAR DSP platform [8] and runs on the Dutch GRID infrastructure. Implementation of the DD pipeline within the same framework is ongoing and will be presented in the future.

3.2 Direction Independent Pipeline

The LOFAR telescope consists of many stations (clusters of electronically coupled antennas), each with an independent electronic gain. The station-based gain calibration parameters can be deduced from the observation of a bright calibrator source before or after the science target [29]. This calculation is performed by the calibration step of the DI pipeline (Fig1). The results from this step are applied to the science target, which is then averaged and processed. This includes removing Radio Frequency Interference and subtraction of bright off-axis sources, and finally calibration against a skymodel derived from other radio surveys [6] [7] [29]. These steps are performed by the Target step of pre-FACTOR [9]. The result is a DI-calibrated dataset which is up to 64 times smaller than the uncalibrated archived data.

The Direction Independent pipeline (Fig.2) consists of an existing set of scripts which use the LOFAR software suite [30] and pre-FACTOR [9] to process the archived data. A parameter-set file (parset) defines a sequence of procedures and their corresponding input parameters. Each procedure may launch one or more executables.

3.3 DI Data Flow and GRID Implementation

The LoTSS survey is led by Leiden University and conducted by a large international group of scientists. Most of the host institutions of those scientists do not have a dedicated network connection to the LOFAR data archive. This means they must download archived (16TB per dataset) data over a public connection. In the case of Leiden, the sustained download rate is 10 to 30 MB/s. This is too low to download a full-size archived dataset in a reasonable time. Downloading of one dataset completes in two weeks, 10 times longer than the DI processing. At this rate, transferring 3000 datasets would take up to a century. This download bottleneck was already recognized by the LOFAR spectroscopy group (PI Oonk) who developed the LOFAR DSP platform [8] for large-scale processing.

To mitigate download issues, the processing was moved to the SURFsara compute grid location at the Amsterdam Science Park[2] as there were previous successes in processing LOFAR data at SURFsara by the LOFAR Spectroscopy group [8]. The pre-FACTOR package runs within the generic pipeline framework[3] which is part of the LOFAR software stack [30]. This framework cannot run on the SURFsara Gina cluster [31] out of the box. It requires either a mounted shared file system or node to node communication, and the Gina nodes offer neither. Work was done to implement the current pre-FACTOR package on the existing LOFAR DSP platform [8]. This work resulted in the LRT framework presented here: a package allowing the implementation of different LOFAR processing pipelines on a distributed infrastructure.

In the case of pre-FACTOR, the two steps of the DI reduction, the Calibrator and Target, were each split in two parts (Fig.2). The first parts of the Calibrator and Target processing are parallelized by running one subband per node, resulting in 244 concurrent jobs. This takes advantage of the data level parallelism of initial LOFAR processing. Running 244 concurrent jobs is also a natural way to process the data as each observation is stored in 244 individual files (as discussed in Section 3.1).

Additionally, splitting the computation makes it more robust. In the case that the download or processing of one job fails, it can be restarted without disrupting parallel jobs. When a step has finished processing (for instance, the Calibration step in Fig.1), the next step can be launched automatically enabling the massive processing of LOFAR Surveys data.

The pre-FACTOR software was designed to be run on single node or clusters with a shared file system. Because the worker nodes at the SURFsara cluster have isolated storage, scripts are included in the LOFAR Reduction Tools to load the relevant data on the worker node before processing. After a job is finished, the scripts save intermediate results to an external storage location [8].

Using intermediate storage to hold the results from each step, the pre-FACTOR DI pipeline was split into four steps as shown in Fig. 2. The Calibrator 1 and Target 1 steps download the raw data at one file per worker node and store the processing results (Calibration Tables and Processed data respectively) in storage. After all Calibrator 1 jobs finish, the Calibrator 2 step combines the results produced into a single calibration table. This table is then applied to the science target by the Target 1 jobs. Finally the Target 2 job combines 10 datasets produced by Target 1 and creates the final DI calibrated datasets.

[2] http://docs.surfsaralabs.nl/projects/grid/en/latest/Pages/Service/system_specs.html
[3] http://www.astron.nl/citt/genericpipeline/

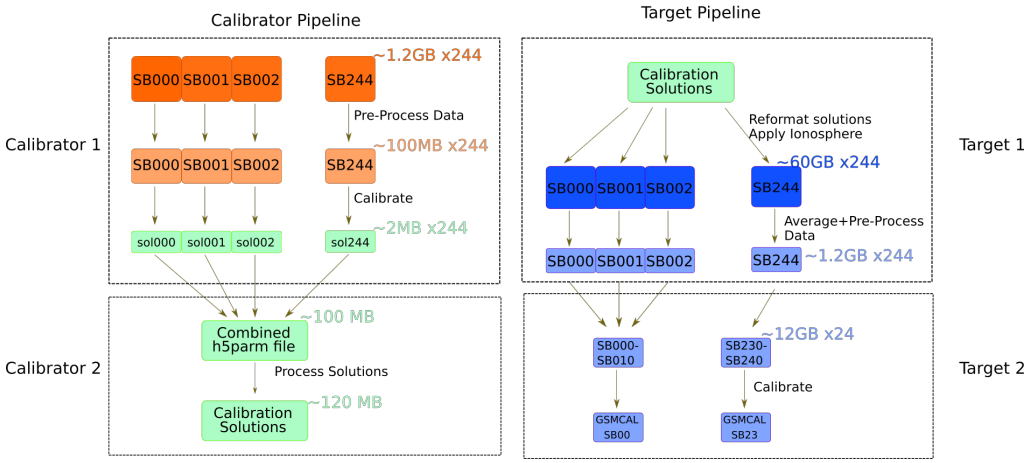


Figure 2: Data flow and parallelization of the Direction Independent Processing. The Calibrator 1 and Target 1 steps run concurrently as independent jobs. Calibrator 2 and Target 2 combine these results. Note that the Target 1 step requires the solutions produced by the Calibrator 2 step. This places a strict ordering on the processing steps.

4. Framework Design

The LRT framework (Fig. 4) was developed to automate the LOFAR Direction Independent calibration by processing the data on the Gina cluster at SURFsara [31]. It is built on the LOFAR DSP platform [8], which facilitates building distributed computing frameworks. A component diagram is shown in Fig.3.

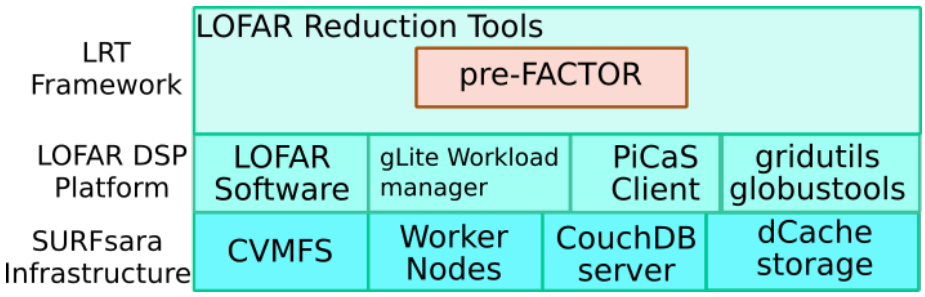


Figure 3: Schematic of the dependencies of the LRT framework, the modules of the LOFAR DSP platform [8], and the infrastructure provided by SURFsara.

By making each job self-contained, the LRT framework provides a portable way to execute the LOFAR scripts (Section 4.3). The Gina cluster provides more than 6000 cores over 300 processing nodes. To take advantage of these capabilities, the framework was made scalable (Section 4.4). Typical processing regularly scales out to 244 jobs per step. To process the LoTSS observations efficiently, automation was built into the LRT tools (Section 4.6). Finally, the framework is con-

structed to allow the inclusion of different processing suites, building onto the generality provided by the LOFAR DSP [8] platform. This generality is now used by several1 LOFAR projects (Section 4.7).

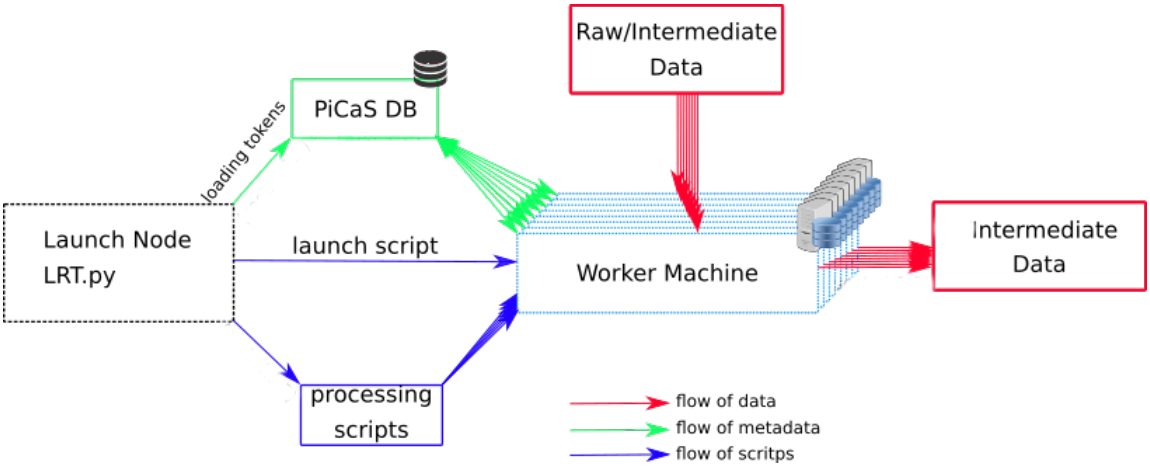


Figure 4: Overview of the design of the LRT framework. Shown is the decoupling of scripts from metadata, compute elements and data storage.

4.1 Framework Elements

The LRT framework consists of a set of modules responsible for different parts of the data reduction. The `srmlist` module handles the storage links to the archived data. If the data is on tape, it sends a command to transfer it to disk. The `sandbox` module packs the processing scripts in an archive and uploads it to disk storage on a dCache system[4] [32]. Doing so makes the processing portable. The `Token` module is responsible for managing metadata, creating job tokens which define a processing job. As the tokens can store arbitrary data, the LRT modules are easily generalizable to other workflows. Documentation of these modules and examples can be found on the LRT github page[5].

4.2 Framework Capabilities

The LRT implementation is designed to be platform independent. It allows for easy extension, thereby enabling LOFAR reduction schemes other than the LoTSS reduction. Two examples of this are the updated LOFAR GRID spectroscopy and LOFAR GRID pre-processing pipelines [8]. We will describe how we use the LOFAR DSP platform to achieve scalability, portability, automation and generality.

4.3 Portability

The Gina architecture requires processing scripts to be stored remotely from worker nodes. These scripts are archived and uploaded to distributed storage and their location is added to the

[4] http://docs.surfsaralabs.nl/projects/grid/en/latest/Pages/Service/system_specifications/dcache_specs.html
[5] https://github.com/apmechev/GRID_LRT

job description. After a worker node locks a job token, it downloads and extracts the processing scripts, reads the metadata from the token and begins the processing (Fig.5).

Storing the scripts location in the job description allows different steps of the pipeline to use different sets of scripts. The benefit from this design is that as long as the worker node has access to the Universal Resource Identifier (URI) of the scripts, it can process the data. This choice provides portability, enabling processing over a variety of distributed computing environments, including sites in the the European Grid Infrastructure [34].

The PiCaS tokens can store strings and integer values as well as file attachments. The LRT DI pipeline implementation uses attachments to store diagnostic files used to assess data quality, lists of links to the data, and parset files that define the pre-FACTOR workflow. Storing these files in a central database means any worker node can read this data at runtime, regardless of the node's location.

The pre-FACTOR scripts require an installation of the LOFAR software stack [35]. These requirements are met by mounting a CVMFS [18] [36] installation of the LOFAR software stack. The CVMFS service provides a portable pre-compiled copy of the LOFAR software. With the CVMFS prerequisite satisfied and an active grid proxy, any computer can download the data and participate in any data reduction step.

4.4 Scalability

The LRT framework can define, launch and monitor jobs on a distributed computing infrastructure. As such, it is effective for pipelines that independently process large datasets in parallel. Each part of the dataset is processed on a single node, and the metadata of this job is stored and updated in a remote database which can be read from and written to by the worker node. A schematic of the communication between worker nodes and the PiCaS database is in Fig.6.

By using a concurrent document-oriented database such as CouchDB [19], each document can store the metadata describing a single processing job. This is not possible with relational databases such as MySQL [33]. These documents are called Tokens, as defined by the PiCaS framework [17]. Processing is scaled out by creating the required number of jobs and launching them on independent nodes. This system can easily scale to tens of thousands of jobs, and currently stores the metadata of thousands of LOFAR jobs.

The first implementation of PiCaS and CouchDB in the LOFAR-DSP platform was carried out in the context of the LOFAR spectroscopy project and custom user processing [8]. This first implementation focused on processing individual data sets and required a high-level of user interaction. Here we build upon the LOFAR DSP platform by making it easier to define the structure of tokens in a text file. Additionally, we provide the automation to process multiple runs (calibrator and target) and handle their products.

4.5 Intermediate Data Storage

Splitting the processing into multiple steps requires intermediate data to be stored at a location accessible to the worker nodes. As the current processing is done at the Gina cluster, the intermediate results are stored in several dedicated storage pools hosted by SURFsara. At each step, the LRT processing scripts check whether the required input data exists and downloads it. This avoids

unnecessary repetition of reduction steps and allows to restart processing from the point of failure rather than from the start. Since the PiCaS tokens can hold the location of the intermediate data, processing becomes scalable to any location that has access to the intermediate data. For example, the final DI datasets are used by DD calibration at several institutes in the Netherlands and Europe.

4.6 Automation

Running many jobs in parallel as part of a multi-step workflow requires automatically launching and restarting steps. Because of the strict ordering of the pipeline steps, each step must wait for the previous step to complete. A PiCaS query is used to tally the number of completed jobs in a step. Once a threshold is reached, the next step automatically launches. Since PiCaS stores the state of each job, failed runs can be restarted automatically by a script which monitors the status of jobs in the database.

Creating tokens is also automated. The user can define the structure of their job token in a text file, and use that file to automatically create tokens. This allows easy and fast creation of tokens holding different sets of metadata. Similarly, the scripts destined for a worker node are packed in an archive called a 'sandbox'. The list of scripts and repositories stored in this archive are stored in a text file, allowing to automatically create different sandboxes by changing this configuration file. A user only has to specify the ID of the observation they're interested in, and the list of files they need to process before launching the DI pipeline.

4.7 Generality

A PiCaS token can hold arbitrary metadata and store configuration files. A user can define their workflow by deciding on the set of steps and the processing done at each step. Once each step launches, it can read the metadata it requires from the Token and load the required software from the CVMFS server described in Section 4.3. Other pipelines can be combined with the LRT framework. This is done by defining the necessary token fields and specifying the processing scripts of the pipeline steps.

While this work discusses the implementation of the LOFAR DI pipeline, work is ongoing to also port the LOFAR Direction Dependent pipeline on the LRT framework.

4.8 LOFAR LoTSS Use Case

LOFAR observations are split and stored in frequency chunks called subbands. A LoTSS observation consists of 244 subbands for the calibrator and 244 subbands for the target. The Calibrator 1 and Target 1 step of the DI Pipeline processes these subbands independently. This is a form of data-level parallelism and increases data throughput.

Without a framework to automate and distribute the processing and a cluster at an LTA location, these datasets would need to be downloaded to an institute's cluster. Such standalone runs of the pre-FACTOR scripts typically process one observation in two weeks dominated by the data transfer time. At the 10-30MB/s connection (the sustained speed at Leiden University), the downloading would take between 30 and 100 years. At SURFsara, the 1Gbps external connection is fast enough to download and process the 16TB in a day and a half. While clusters at typical institutions number in the tens of nodes, the Gina cluster at SURFsara has over 300 nodes. Each of the 244

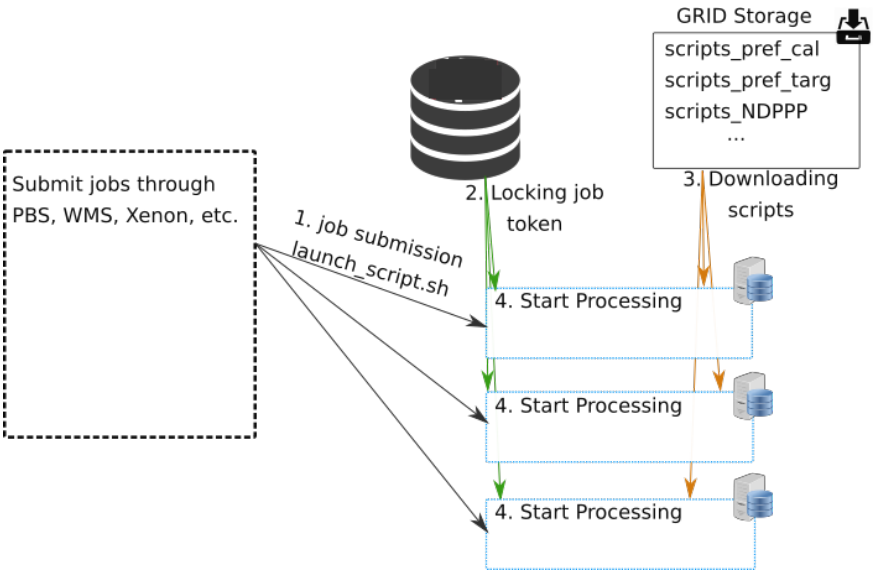


Figure 5: Starting processing on worker nodes. Currently processing is done on SURFsara Gina cluster , however the framework has been tested at the Leiden University cluster.

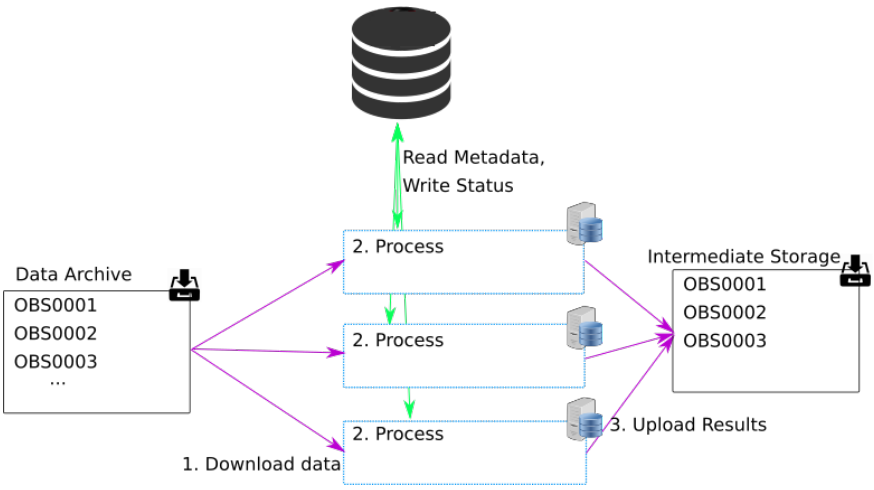


Figure 6: Processing of LOFAR data from the Long Term Archive with results stored at an intermediate storage location.

subbands is run on a dedicated node concurrently. This massive parallelization further increases data throughput.

Porting the LOFAR LoTSS data reduction to the SURFsara Gina cluster using the LRT package has resulted in a 15x increase in throughput. Suggestions on further increasing the amount of data processed are presented in Section 5.1.

5. Conclusion and Future Work

The goal of the LRT framework is to build upon the LOFAR DSP platform to create a package to port LOFAR processing to a massively distributed compute environment. The LRT framework was designed to be scalable, portable, automated and general. The DI pipeline of the LoTSS survey was used as a demonstration of the capabilities of the LRT software.

Combining the DI pipeline scripts with the LRT tools resulted in a 15x increase in throughput compared to previous LoTSS data reduction strategies which were dominated by data transfer. Automation was provided by separating the different parts of execution into separate modules and using configuration files to facilitate creating workflows. Thanks to this automation, it is possible to perform the processing necessary for the LoTSS survey.

The portability of the LRT framework makes it easy to move processing to other compute locations such as those near the data archive, increasing the throughput. This portability is provided by installing the software on a CVMFS server that worker nodes can access and by storing metadata on an external PiCaS server. Separating scripts and metadata from the computation elements makes it possible to use computational resources at multiple sites as required.

The scalability of this framework allows to launch multiple data reductions concurrently and easily monitor progress through a web-accessible CouchDB interface. Scalability is achieved by storing metadata in a document based database with asynchronous write support, and by running each job on an isolated node.

The framework is made general by using PiCaS tokens, capable of storing arbitrary metadata, and passing this data to the processing scripts. Additionally, by separating the processing from the data retrieval, the framework can 'plug-and-play' different software and execute it on the same dataset. Finally, as the framework is general, other LOFAR projects can benefit from incorporating their processing into the LRT framework.

Using the LRT framework, more than 100 datasets have passed through the Direction Independent pipeline. This corresponds to a throughput of $\sim$1 dataset per day. Future improvements (Section 5.1) are expected to increase the throughput to two datasets per day. This is the minimum rate required by the LoTSS survey.

5.1 Future Work

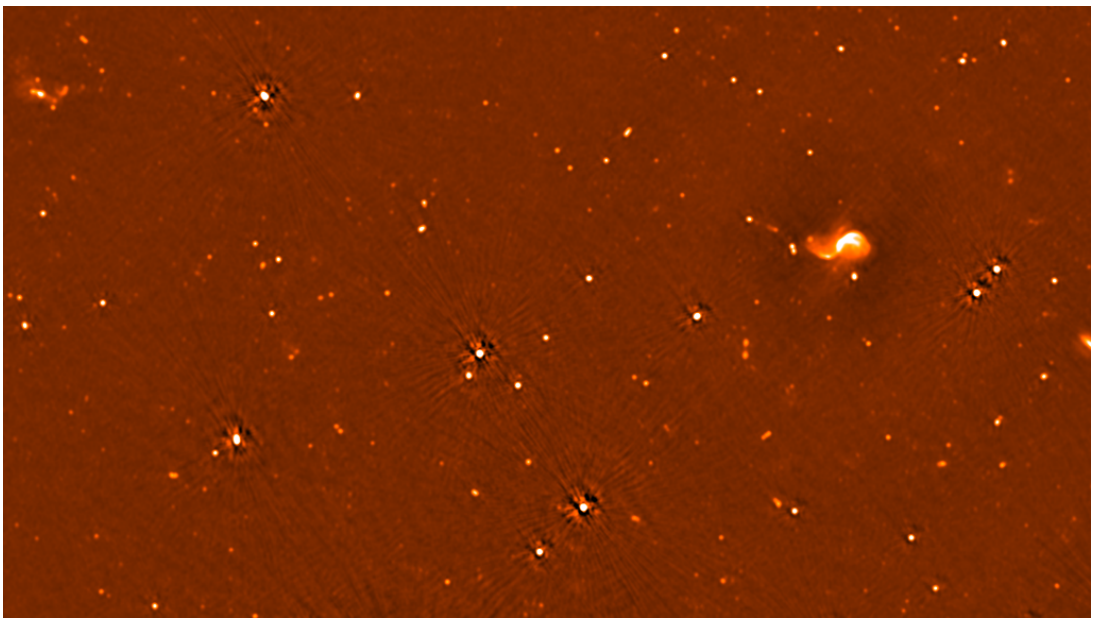
A significant portion of the LoTSS data is stored at the FZ Jülich data centre[6]. Because of the high data sizes, even the 1 Gbps transfer between this site and SURFsara is insufficient to process two observations per day. We are currently investigating porting the LRT framework to the FZJ site as well. Doing so will reduce the data size by a factor of 64. This will make transfer to other processing locations possible within a few hours for each dataset.

While the LRT framework successfully automated the Direction Independent calibration pipeline, it still needs to implement the Direction Dependent processing scripts.

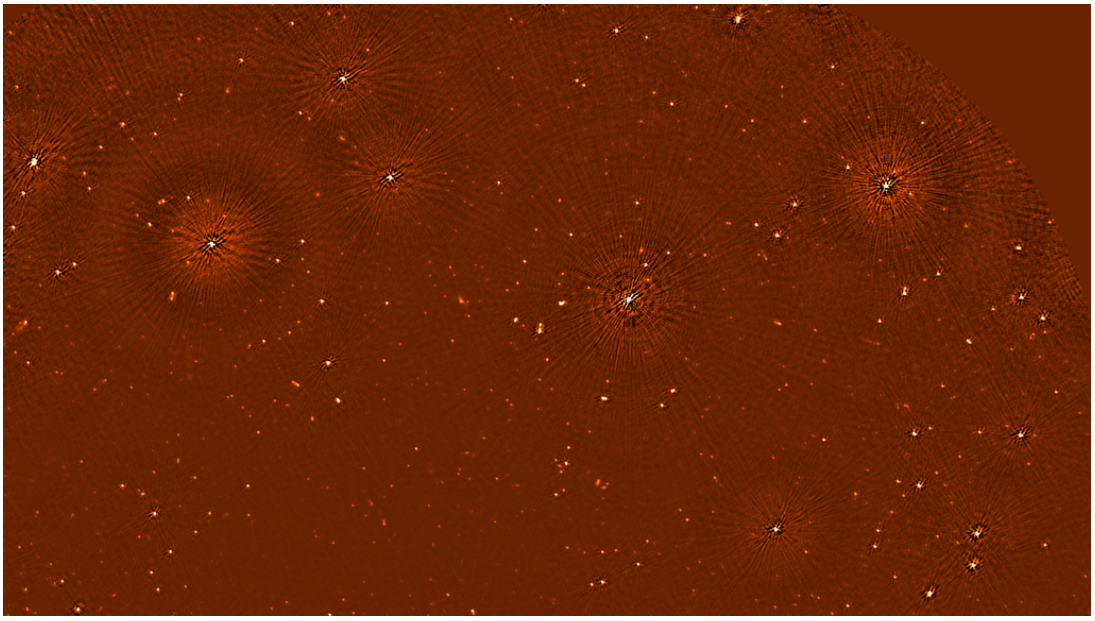
DIRAC [37] or Xenon [38] are two middleware packages that allow launching jobs at multiple clusters from a single location. DIRAC is expected to replace the current workload management system at EGI sites, and Xenon is used by some projects at SURFsara [21]. Due to the portability of

[6] http://www.fz-juelich.de/portal/EN/Home/home_node.html

the LRT software, the LOFAR processing can be launched on other clusters using such middleware. Launching LRT jobs at other LTA sites will reduce the size of archived data so it can be transported faster.



(a) Field centred on 12:15:00,+47:00:00 showing M106 on the right.



(b) Image of a field centred on 12:22:00,+53:40:00 produced by the DI pipeline. Artefacts around the bright point sources are to be removed by the DD pipeline.

Figure 7: A sample of LoTSS image, after Direction Independent calibration with pre-FACTOR using the LRT framework. Artifacts such those in Fig. 7b need to be removed by Direction Dependent processing (Section 3.1 , Figure 1)

Finally, as the reduction is automated, it can in principle be started shortly after the telescope finishes the observation. Launching jobs immediately after an observation will minimize the overhead spent moving the data from tape to disk, which can take up to a week.

Minimizing the latency between observation and science quality images will benefit the LOFAR community immensely. This fast turnover will allow radio astronomers to focus on their specific science case. An all-sky survey at the 150 MHz range will create a multitude of targets for follow-up with optical telescopes such as the LOFAR-WEAVE survey [39]. Figure 7 shows a small sample of interesting objects in the data processed using the software presented in this publication. A full list of science results expected from the LoTSS project can be found in [6]. Efficient high-throughput processing of LOFAR data will empower these science cases opening the way to exciting new discoveries.

Acknowledgement

This work was carried out on the Dutch national e-infrastructure with the support of SURF Cooperative through grant e-infra 160022 & 160152.

References

[1] MP Van Haarlem, MW Wise, AW Gunst, George Heald, JP McKean, JWT Hessels, AG De Bruyn, Ronald Nijboer, John Swinbank, Richard Fallows, et al. Lofar: The low-frequency array. *Astronomy & Astrophysics*, 556:A2, 2013.

[2] Vibor Jelić, Saleem Zaroubi, Panagiotis Labropoulos, Rajat M Thomas, Gianni Bernardi, Michiel A Brentjens, AG De Bruyn, Benedetta Ciardi, Geraint Harker, Leon VE Koopmans, et al. Foreground simulations for the lofar–epoch of reionization experiment. *Monthly Notices of the Royal Astronomical Society*, 389(3):1319–1335, 2008.

[3] PEF Dewdney, TJW Lazio, PJ Hall, and RT Schilizzi. The square kilometer array (ska) radio telescope: Progress and technical directions. *Radio Science Bulletin*, 326:4–19, 2008.

[4] Hanno Holties, Adriaan Renting, and Yan Grange. The lofar long-term archive: e-infrastructure on petabyte scale. In *SPIE Astronomical Telescopes+ Instrumentation*, pages 845117–845117. International Society for Optics and Photonics, 2012.

[5] Domingos Barbosa, João Paulo Barraca, Dalmiro Maia, Bruno Carvalho, Jorge Vieira, Paul Swart, Gerhard Le Roux, Swaminathan Natarajan, Arnold van Ardenne, and Luis Seca. Power monitoring and control for large scale projects: Ska, a case study. In *SPIE Astronomical Telescopes+ Instrumentation*, pages 99100L–99100L. International Society for Optics and Photonics, 2016.

[6] TW Shimwell, HJA Röttgering, PN Best, WL Williams, TJ Dijkema, F de Gasperin, and MJ Hardcastle. The lofar two-metre sky survey. i. survey description and preliminary data relase. *Astronomy & Astrophysics*, 2016.

[7] RJ Van Weeren, WL Williams, MJ Hardcastle, TW Shimwell, DA Rafferty, J Sabater, G Heald, SS Sridhar, TJ Dijkema, G Brunetti, et al. Lofar facet calibration. *The Astrophysical Journal Supplement Series*, 223(1):2, 2016.

[8] J.B.R. Oonk, A.P. Mechev, A. Danezi, C. Schrijvers, T.W. Shimwell Radio astronomy on a distributed shared computing platform: The LOFAR case. In preparation 2017.

[9] Horneffer A. Prefactor: Pre-facet calibration pipeline. https://github.com/lofar-astron/prefactor, 2017.

[10] Nick Kaiser. Moore's law takes on the universe; new astronomy with giga-pixel imagers and peta-byte data archives. In *Aerospace conference, 2009 IEEE*, pages 1–2. IEEE, 2009.

[11] Shayan Shams, Nayong Kim, Xiandong Meng, Ming Tai Ha, Shantenu Jha, Zhong Wang, and Joohyun Kim. A scalable pipeline for transcriptome profiling tasks with on-demand computing clouds. In *Parallel and Distributed Processing Symposium Workshops, 2016 IEEE International*, pages 443–452. IEEE, 2016.

[12] Anjani Ragothaman, Sairam Chowdary Boddu, Nayong Kim, Wei Feinstein, Michal Brylinski, Shantenu Jha, and Joohyun Kim. Developing ethread pipeline using saga-pilot abstraction for large-scale structural bioinformatics. *BioMed research international*, 2014, 2014.

[13] JD Van Horn, J Dobson, J Woodward, M Wilde, Y Zhao, J Voeckler, and I Foster. Grid-based computing and the future of neuroscience computation, methods in mind, 2005.

[14] William K Michener and Matthew B Jones. Ecoinformatics: supporting ecology as a data-intensive science. *Trends in ecology & evolution*, 27(2):85–93, 2012.

[15] Ewa Deelman, Gurmeet Singh, Mei-Hui Su, James Blythe, Yolanda Gil, Carl Kesselman, Gaurang Mehta, Karan Vahi, G Bruce Berriman, John Good, et al. Pegasus: A framework for mapping complex scientific workflows onto distributed systems. *Scientific Programming*, 13(3):219–237, 2005.

[16] Yong Zhao, Mihael Hategan, Ben Clifford, Ian Foster, Gregor Von Laszewski, Veronika Nefedova, Ioan Raicu, Tiberiu Stef-Praun, and Michael Wilde. Swift: Fast, reliable, loosely coupled parallel computation. In *Services, 2007 IEEE Congress on*, pages 199–206. IEEE, 2007.

[17] Picas overview - grid documentation v1.0. http://doc.grid.surfsara.nl/en/latest/Pages/Practices/picas/picas_overview.html, 2017.

[18] C Aguado Sanchez, J Bloomer, P Buncic, L Franco, S Klemer, and P Mato. Cvmfs-a file system for the cernvm virtual appliance. In *Proceedings of XII Advanced Computing and Analysis Techniques in Physics Research*, volume 1, page 52, 2008.

[19] Chirs ANDERSON. Apache couchdb: The definitive guide. *http://couchdb. Apache. org/index. htm Acessado em*, 5(06):2009, 2009.

[20] Jan Bot. Picas: Python client using couchdb as a token pool server. https://github.com/jjbot/picasclient, 2017.

[21] Joris Borgdorff, Harsha Krishna, and Michael H Lees. Sim-city: An e-science framework for urban assisted decision support. *Procedia Computer Science*, 51:2327–2336, 2015.

[22] Y Li. Reliability of long heterogeneous slopes in 3d: Model performance and conditional simulation. 2017.

[23] Marco Clemencic and B Couturier. A new nightly build system for lhcb. In *Journal of Physics: Conference Series*, volume 513, page 052007. IOP Publishing, 2014.

[24] Tom SANTE. Development of (graphical) web applications for the processing and interpretation of arraycgh data. 2010.

[25] Grigory Rybkin. Atlas software packaging. In *Journal of Physics: Conference Series*, volume 396, page 052063. IOP Publishing, 2012.

[26] GS Davies, JP Davies, Brian Rebel, Kanika Sachdev, Jan Zirnstein, C Group, et al. Software management for the noνaexperiment. In *Journal of Physics: Conference Series*, volume 664, page 062011. IOP Publishing, 2015.

[27] GA Renting and HA Holties. Lofar long term archive. In *Astronomical Data Analysis Software and Systems XX*, volume 442, page 49, 2011.

[28] WN Brouw. Aperture synthesis. In *Image Processing Techniques in Astronomy*, pages 301–307. Springer, 1975. The Astrophysical Journal Supplement Series, 223(1):2, 2016.

[29] Cyril Tasse, Ger van Diepen, Sebastiaan van der Tol, Reinout J van Weeren, Joris E van Zwieten, Fabien Batejat, Sanjay Bhatnagar, Ilse van Bemmel, Laura Bîrzan, Annalisa Bonafede, et al. Lofar calibration and wide-field imaging. *Comptes Rendus Physique*, 13(1):28–32, 2012.

[30] RF Pizzo et al. The lofar imaging cookbook v2.0. *internal ASTRON report*, 2010.

[31] Gina specifications - grid documentation v1.0. Available at http://docs.surfsaralabs.nl/projects/grid/en/latest/Pages/Service/system_specifications/gina_specs.html, 2017.

[32] Patrick Fuhrmann and Volker Gülzow. dcache, storage system for the future. In *European Conference on Parallel Processing*, pages 1106–1113. Springer, 2006.

[33] Randy Jay Yarger, George Reese, Tim King, and Andy Oram. *MySQL and mSQL*. O'Reilly & Associates, Inc., 1999.

[34] EGI: The European Grid Infrastructure https://www.egi.eu/, 2017.

[35] Lofar software stack. http://www.lofar.org/wiki/doku.php?id=public:software_stack_installation, 2017.

[36] Softdrive on the grid. Available at http://docs.surfsaralabs.nl/projects/grid/en/latest/Pages/Advanced/grid_software.html#softdrive .

[37] Andrei Tsaregorodtsev, Vincent Garonne, and Ian Stokes-Rees. Dirac: A scalable lightweight architecture for high throughput computing. In *Grid Computing, 2004. Proceedings. Fifth IEEE/ACM International Workshop on*, pages 19–25. IEEE, 2004.

[38] Jason Maassen, Stefan Verhoeven, Joris Borgdorff, Niels Drost, cwmeijer, Jurriaan H. Spaaks, Rob V. van Nieuwpoort, Piter T. de Boer, and Ben van Werkhoven. Xenon: Xenon 1.1.0, December 2015.

[39] Gavin Dalton, Scott C Trager, Don Carlos Abrams, David Carter, Piercarlo Bonifacio, J Alfonso L Aguerri, Mike MacIntosh, Chris Evans, Ian Lewis, Ramon Navarro, et al. Weave: the next generation wide-field spectroscopy facility for the william herschel telescope. In *SPIE Astronomical Telescopes+ Instrumentation*, pages 84460P–84460P. International Society for Optics and Photonics, 2012.