PROCEEDINGS OF SCIENCE

FACT-Tools – Streamed Real-Time Data Analysis

K. A. Brügge[b]*, M. L. Ahnen[a], M. Balbo[c], M. Bergmann[d], A. Biland[a], C. Bockermann[e], T. Bretz[a], J. Buss[b], D. Dorner[d], S. Einecke[b], J. Freiwald[b], C. Hempfling[d], D. Hildebrand[a], G. Hughes[a], W. Lustermann[a], K. Mannheim[d], K. Meier[d], K. Morik[e], S. Müller[a], D. Neise[a], A. Neronov[c], M. Nöthe[b], A.-K. Overkemping[b], A. Paravac[d], F. Pauss[a], W. Rhode[b], F. Temme[b], J. Thaele[b], S. Toscano[c], P. Vogler[a], R. Walter[c], and A. Wilbert[d]

Email: kai.bruegge@tu-dortmund.de

[a] *ETH Zurich, Institute for Particle Physics*
Otto-Stern-Weg 5, 8093 Zurich, Switzerland
[b] *TU Dortmund, Experimental Physics 5*
Otto-Hahn-Str. 4, 44221 Dortmund, Germany
[c] *University of Geneva, ISDC Data Center for Astrophysics*
Chemin d'Ecogia 16, 1290 Versoix, Switzerland
[d] *Universität Würzburg, Institute for Theoretical Physics and Astrophysics*
Emil-Fischer-Str. 31, 97074 Würzburg, Germany
[e] *TU Dortmund, Lehrstuhl für künstliche Intelligenz*
Otto-Hahn-Str. 12, 44221 Dortmund, Germany

The First G-APD Cherenkov telescope (FACT) is dedicated to monitor bright TeV blazars in the northern sky. The use of silicon photon detectors allows for a larger duty cycle, which results in a huge amount of collected data (800 GB/night). In order to satisfy its monitoring purpose, changes in the flux of the observed sources have to be registered without delay. This requires a data analysis that provides physical results at a rate that is comparable to the trigger rate of 80Hz. The recently developed data analysis software FACT-Tools aims to accomplish these requirements in real-time. It is implemented based on of the `streams`-framework, which was developed at Dortmund's collaborative research center for resource-constrained data analysis (SFB 876). `Streams` delivers an easy-to-use abstraction layer to design analysis processes by use of human readable XML files, which also make the analysis reproducible. Multi-source processes (e.g. simultaneous analyses of data from several telescopes) and multi-core processes (parallelization) are already included in the `streams`-framework. Therefore, `Streams` is an ideal framework for use in gamma-ray astronomy.

The FACT-Tools are an extension library for the `streams`-framework with analysis methods for Cherenkov telescopes. The collection of methods is ranging from RAW data handling and calibration up to image parameter extraction and Gamma-Proton classification. The latter is performed by an online application of a random forest classifier, which in turn, allows for an adaptation in other tasks e.g. image cleaning or online estimation of the energy spectrum.

In this contribution we want to present the features of FACT-Tools and the `streams`-framework alongside with their performance measured on the data from the FACT Cherenkov telescope.

The 34th International Cosmic Ray Conference,
30 July – 6 August, 2015
The Hague, The Netherlands

1. Introduction

FACT [1], the **F**irst G-**A**PD **C**herenkov **T**elescope, is an Imaging Air Cherenkov Telescope, located on the Canary Island of La Palma. FACT uses Silicon Photomultipliers (SiPMs) instead of conventional Photomultiplier Tubes to detect Cherenkov photons, produced by charged, secondary particles in extensive air showers.

Its main purpose is the continuous monitoring of bright TeV gamma ray sources in the Northern Hemisphere. The reflector is made up of 30 hexagonal mirrors with a total reflective area of 9.51 m^2. The FACT camera consists of 1440 SiPM pixels each with with a maximum sampling rate of up to 2 Gigasamples. The samples are read by a DRS4 (Domino Ring Sampler) chip. In case an event is triggered, 300 samples corresponding to 150 ns are stored directly to hard drives at the experiment site. This creates about 900 kB of data per recorded Event. An average trigger rate of up to 80 Hz leads to roughly 800 GB of data each night.

The central task of a monitoring system is to detect changes in gamma-ray flux from the source and to alert other experiments if some predefined limit is reached. The currently running system calculates excess rates with some delay after the data was written to hard disk [2]. Our goal is to run the entire data analysis process to run under the strong time and memory constraints given at the experiment site. In order to calculate the flux of an observed source we need to run an entire data analysis process including calibration, image cleaning, parametrization and Gamma-Proton classification. Our goal is to run the whole data analysis process in near real time.

An analysis software called FACT-Tools [3] built on top of the `streams`-framework [4] has been implemented. Utilizing the features of streamed data analysis, the goals mentioned above can be achieved.

2. The `streams`-framework

The data analysis chain for the FACT telescope, or really any other experiment, can be modeled by a data flow graph. A data flow graph consists of different types of interconnected nodes representing single steps of the analysis chain. The flow graph contains a node representing the source of the data we want to analyze. The source continuously emits pieces of data, called data items, into the graph for processing. These data items can be transferred along the edges of the graph to nodes processing the data in some appropriate way. These nodes can then pass the, possibly modified, data item along to the next node.

The main motivation behind the `streams`-framework is to provide a light-weight API to uniquely define the topology of data flow graphs using an XML based description language. This allows for simple reproducibility of your data analysis while also providing a thin abstraction layer for your data flow. This allows for robust reasoning about the data and control flow of your data analysis chain. The `streams`-framework is written in the Java programming language and comes with a runtime module to execute the stream as defined by the XML document. These data flow graphs can also be easily mapped to topologies for the Apache Storm [5] engine or Apache Hadoop clusters. This allows for data analysis in large scale distributed environments.

*Corresponding Author

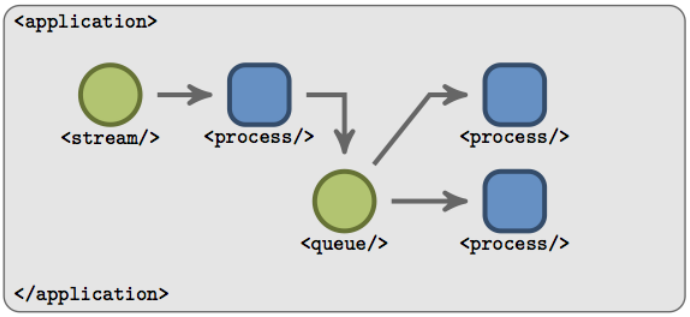

Figure 1: A graphical representation of a data flow graph as defined by an XML file adhering to the naming convention of the `streams`-framework

In the context of the `streams`-framework a node processing some data is called a *processor*. Nodes that act as a source for data items are called *streams*. Within the graph data streams can be collected and distributed using *queues*. Each data item created by the stream is represented by a HashMap containing values of arbitrary type. The `streams`-framework comes with a set of basic processors for data transformations and control flow management. In addition to the already existing processors a user can also easily define an entirely new processor. Implementing a Java Interface of type `Processor` suffices. The example code in listing 1 demonstrates how to read and modify some generic data from the stream.

Listing 1: Example code demonstrating how to write a custom processor.

```
public class MyProcessor implements Processor {
  @Override
  public Data process(Data item) {
    double[] data = (double[]) item.get("TelescopeData");
    double[] smoothedData = smoothData(data);
    item.put("SmoothedData", smoothedData);
    return input;
  }
}
```

3. The FACT-Tools as an extension to the `streams`-framework

The FACT-Tools are built on the `streams`-framework by adding methods and algorithms for analyzing FACT specific data. To get physically meaningful results from the raw data recorded by the telescope a handful of important steps have to be performed. At the moment the data acquisition system writes telescope raw data as FITS [6] [7] files. The FACT-Tools provide a stream implementation that can read these files and create one data item per triggered telescope event. The raw data has to be calibrated in order to convert the recorded samples into voltages. In the same step we remove artifacts from the signal. Once the data has been calibrated, two important features are gained from the voltage curves for each pixel. To gain further insight we need to know the number of photons that hit the pixel and their arrival time in the pixel. Both values are estimated from the height and the slope of the voltage curves. In the next step we select only those camera

pixels which were actually hit by any Cherenkov photons from the shower. This step is called image cleaning and is accomplished using relatively simple heuristics. For further investigations we need to calculate image parameters that contain physically relevant information about the nature of the air shower. The so called Hillas parameters belong to the standard set of image parameters for Imaging Atmospheric Cherenkov Telescopes. Once the image parameters are calculated they can be used for separation of the recorded signal into gamma or hadron induced showers. All of these preprocessing methods are included as processors inside the FACT-Tools. Figure 4 is a plot of the runtimes of some of the processors that are executed during the analysis of FACT data. Figure 2 shows the basic elements of the data analysis process. The study of runtime distributions of different processors quickly reveals possibilities for runtime optimizations.

Figure 2: Data processing steps from raw data acquisition to Gamma-Proton classification.

All of these steps can be defined using a single XML file. Listing 2 shows an outline of what a simple data analysis process for the FACT data, including a planned flux estimating step, could look like.

Listing 2: An outline for an XML document describing some data flow.

```
<stream id="fact:data" class="fact.io.FitsStream"
        url="http://sfb876.de/telescopeDataFile.fits"/>

<process input="fact:data">
    <fact.data.DrsCalibration/>
    <fact.data.SignalExtraction />
    <fact.image.ImageCleaning energyThreshold="2.45" />
    <fact.image.features.HillasParameters />
    <streams.weka.Apply modelUrl="http://sfb876.de/modell.mo" />
    <if condition="%{data.@label} == gamma" />
        <fact.flux.Estimate />
</process>
```

The flexible data stream definition and modularity allows for development of new methods in a rapid prototyping fashion. Additionally the FACT-Tools are equipped with a graphical user interface which allows to quickly visualize the results of different preprocessing methods. Figure 3 shows a screenshot of the user interface.

The image parameters and cleaning methods provided by the FACT-Tools achieve good performance in terms of estimator bias [8] and separation strength [9].

4. Online application of machine learning algorithms

Machine learning algorithms can be used to classify telescope events into categories. To use a supervised classifier however it needs labeled training data to build a model with a proper decision

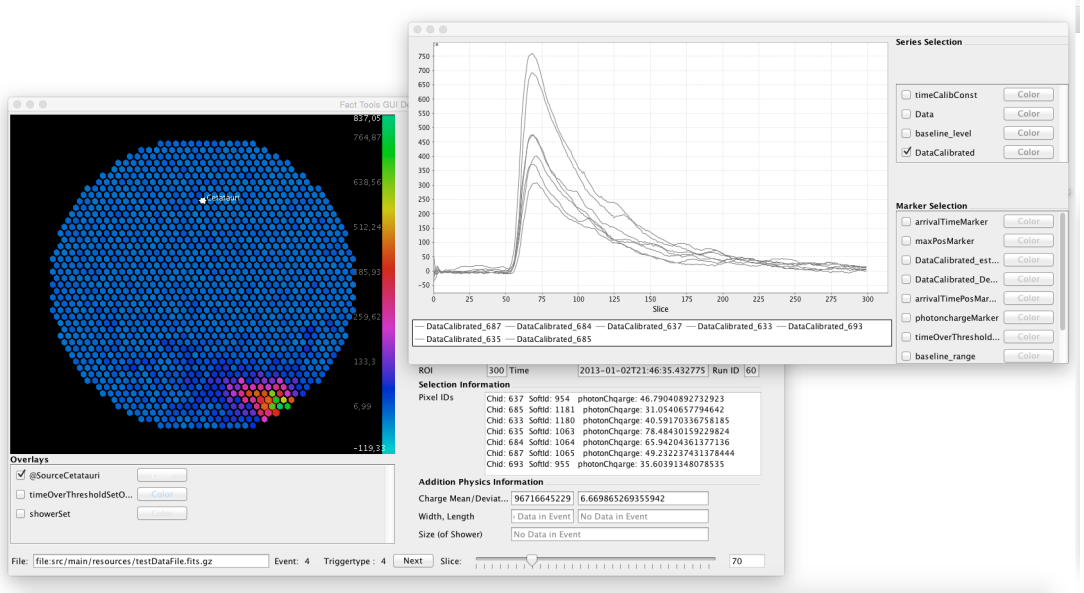

Figure 3: A screenshot of the FACT-Tools graphical user interface

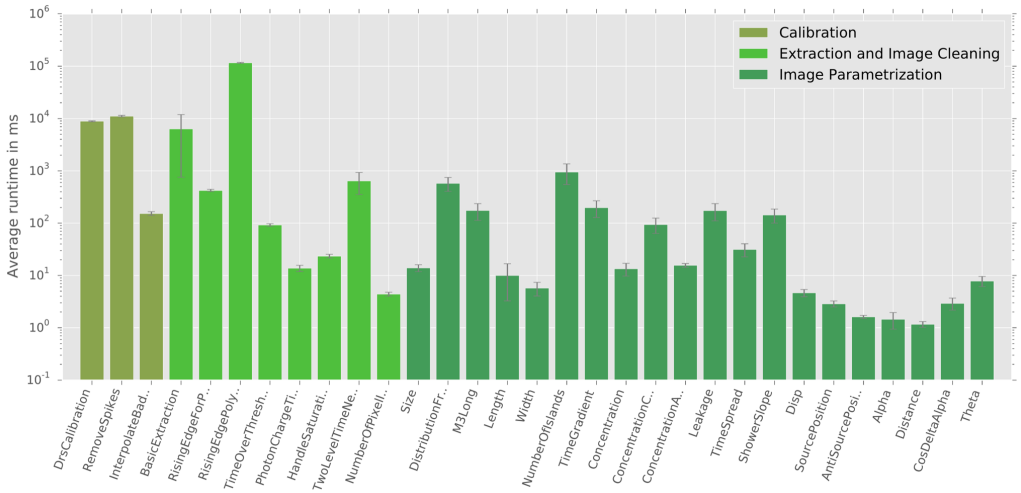

Figure 4: This plot shows the average runtime of different processors as executed during the analysis of FACT data

function. Labeled data can be created using simulated showers from Monte-Carlo simulation programs such as CORSIKA [10] and CERES [11]. These models can be trained in an offline manner since we always have complete sets of simulated data and can produce more if the need arises. Furthermore there is no definite time constraint on training a classifier with Monte-Carlo data. Application of a classifier to real telescope data however should be possible in an online or streaming manner.

The `streams`-framework allows for the integration of popular machine learning libraries such as MOA [12] and WEKA [13]. The model is trained on image parameters gained from simulated

data and then serialized. This serialized model can be used by the *Apply* processor inside a data stream definition for real telescope data. This processor applies the deserialized model to each telescope event and labels it for either class. We use the WEKA Random Forest[14] implementation for separation of gammas and hadrons because the Random Forest classifier has proven to be useful for this classification problem [9].

Based on the label given by the classifier further processing can take place. In our case we plan to use the events labeled as "gamma" for flux estimation.

5. Streaming from multiple sources in multiple threads

Using the `streams`-framework its possible to analyze data from many source at once using more than one thread. This is a common task for batch processing large amounts of files from a single streams definition. Or for reading data from more than one telescope at once. Each Process, as defined by the `<process></process>` tags in the XML, is executed within its own thread. This makes multithreading straightforward. Listing 3 demonstrates how to read and analyze data from more than one stream at once.

Listing 3: A stream processing two data files with two different processes.

```
<stream id="fact_1" class="fact.io.FitsStream"
        url="http://sfb876.de/telescopeData_1.fits"/>

<stream id="fact_2" class="fact.io.FitsStream"
                url="http://sfb876.de/telescopeData_2.fits"/>

<process input="fact_1">
    ...
</process>
<process input="fact_2">
    ...
</process>
<process input="fact_1">
    ...
</process>
```

For batch processing many files this method would be quite tedious however. Using the `copies` keyword a process can easily be copied many times which makes processing lots of files much easier.

Listing 4: A stream processing many data files using the `copies` keyword.

```
<stream id="fact:${copy.id}" class="fact.io.FitsStream"
        url="http://sfb876.de/telescopeData_${copy.id}.fits"/>

<process input="fact:${copy.id}" copies=4>
    ...
</process>
```

It is also possible to collect data from different sources into queues to merge different data streams into one. This allows for maximizing IO throughput of the system while handling inhomogeneous data streams from different sources. The planned Cherenkov Telescope Array (CTA) is a great use case for the `streams`-framework. In the CTA experiment data from over 100 telescopes will be collected and analyzed. For monitoring purposes data from all of the telescopes needs to be analyzed with minimal delay. Utilizing the features of streams the CTA data can be managed and analyzed on distributed and scalable computing infrastructures such as Hadoop or Apache Storm.

6. Conclusions

The FACT-Tools software can analyze FACT telescope data and perform Gamma-Proton classification under real-time constraints using online application of machine learning models. The underlying `streams`-framework provides the tools and methods for distributed and parallel processing of data streams from multiples sources at once. This allows for the creation of large scale analysis chains for experiments using multiple data sources like CTA. In the near future we plan to use the online application of pre-trained models for flux estimation at La Palma, to compare it to the existing quick look analysis [2] and to evaluate these techniques for the planned CTA[15] experiment.

Acknowledgment The important contributions from ETH Zurich grants ETH-10.08-2 and ETH-27.12-1 as well as the funding by the German BMBF (Verbundforschung Astro- und Astroteilchenphysik) and HAP (Helmoltz Alliance for Astroparticle Physics) are gratefully acknowledged. We are thankful for the very valuable contributions from E. Lorenz, D. Renker and G. Viertel during the early phase of the project. We thank the Instituto de Astrofisica de Canarias allowing us to operate the telescope at the Observatorio del Roque de los Muchachos in La Palma, the Max-Planck-Institut für Physik for providing us with the mount of the former HEGRA CT 3 telescope, and the MAGIC collaboration for their support. Part of this work is supported by Deutsche Forschungsgemeinschaft (DFG) within the Collaborative Research Center SFB 876 "Providing Information by Research-Constrained Analysis", project C3.

References

[1] A. Biland et al. *Calibration and performance of the photon sensor response of FACT — the first G-APD Cherenkov telescope*. In: *JINST* 9.10 (2014), P10012.

[2] D. Dorner et al. *FACT – TeV Flare Alerts Triggering Multi-Wavelength Observations*. In: *these proceedings*. PoS(ICRC2015)704. 2015.

[3] C. Bockermann et al. *Online Analysis of High-Volume Data Streams in Astroparticle Physics*. In: *Machine Learning: ECML 2015, Industrial Track*. Springer Berlin Heidelberg, 2015.

[4] C. Bockermann and H. Blom. *The Streams Framework*. Tech. rep. 5. TU Dortmund, Dec. 2012.

[5] *Apache Storm*. July 2015. URL: https://storm.apache.org/.

[6] Pence, W. D. et al. *Definition of the Flexible Image Transport System (FITS), version 3.0*. In: *A&A* 524 (2010), A42. DOI: 10.1051/0004-6361/201015362.

[7] M. L. Ahnen et al. *Data compression for the First G-APD Cherenkov Telescope*. In: *ArXiv e-prints* (June 2015). arXiv: 1506.06045 [astro-ph.IM].

[8] J. Buss et al. *FACT – Influence of SiPM Crosstalk on the Performance of an Operating Cherenkov Telescope*. In: *these proceedings*. PoS(ICRC2015)863. 2015.

[9] F. Temme et al. *FACT – First Energy Spectrum from a SiPM Cherenkov Telescope*. In: *these proceedings*. PoS(ICRC2015)707. 2015.

[10] D. Heck et al. *CORSIKA: A Monte Carlo Code to Simulate Extensive Air Showers," Report FZKA 6019, Forschungszentrum Karlsruhe*. Tech. rep. 1998.

[11] T. Bretz and D. Dorner. *MARS - CheObs goes Monte Carlo*. In: *Proceedings of the 31st ICRC*. 2009.

[12] A. Bifet et al. *MOA: Massive Online Analysis*. In: *J. Mach. Learn. Res.* 11 (Aug. 2010), pp. 1601–1604. ISSN: 1532-4435.

[13] M. Hall et al. *The WEKA Data Mining Software: An Update*. In: *SIGKDD Explor. Newsl.* 11.1 (Nov. 2009), pp. 10–18. ISSN: 1931-0145. DOI: 10.1145/1656274.1656278.

[14] L. Breiman. *Random Forests*. English. In: *Machine Learning* 45.1 (2001), pp. 5–32. ISSN: 0885-6125. DOI: 10.1023/A:1010933404324.

[15] M. Actis et al. *Design concepts for the Cherenkov Telescope Array CTA: an advanced facility for ground-based high-energy gamma-ray astronomy*. English. In: *Experimental Astronomy* 32.3 (2011), pp. 193–316. ISSN: 0922-6435. DOI: 10.1007/s10686-011-9247-0.