The Journal of Open Source Software

SHED: Streaming Heterogeneous Event Data Tracking with Provenance

Christopher J. "CJ" Wright[1], **Songsheng Tao**[1], **and Simon J. L. Billinge***[1, 2]

1 Department of Applied Physics and Applied Mathematics, Columbia University, New York, NY 10027 **2** Brookhaven National Laboratory, Upton, NY 11973

DOI: 10.21105/joss.03119

Software

- Review
- Repository
- Archive

Editor: Arfon Smith

Reviewers:

- @remram44

Submitted: 29 October 2020
Published: 17 March 2021

Summary

The SHED package provides a python based software framework for real time analysis of streaming data with provenance. It is highly flexible, allowing analysis pipelines to be built for any variety of experimental data that is arriving in a time series. It allows straightforward serialization of the pipeline, and the analysis results, for storage in databases with provenance information that allows analyses later to be pulled from the databases, adapted if desired, and rerun.

Statement of Need

The accelerating rate of data coming from modern scientific experiments enable completely new classes of experiments to be carried out such as *in situ* and *operando* measurements of time-dependent phenomena. Many current and planned world class experimental facilities such as x-ray synchrotron sources produce mountains of data. Real-time processing of these large volumes of data can be made possible by streaming data processing, where the data is processed and reduced as it is being collected. While many streaming data processing systems exist (Silva et al., 2007)(Davison et al., 2014)(Picard, 2018)(Carbone et al., 2015) few handle the heterogeneity which is prevalent in many experimental environments. For example, x-ray synchrotrons support thousands of users per year doing a wide array of bespoke experiments and measurements. Additionally, few streaming systems track the provenance of the processed data, creating reproducibility and discoverability concerns.

`SHED` is a Python package for handling heterogeneous streaming data and the tracking of the provenance of the produced results. The `SHED` API translates data coming from a streaming experiment into python objects which are passed into data processing pipelines written using the `streamz` library. It expects data in the form of the Bluesky Event Model schema, which is a flexible open-source document model suitable for scientific time-series experiments. The Bluesky event model (Koerner et al., 2020) is used by nearly all of the beamlines at the National Synchrotron Light Source-II (NSLS-II) as well as at other facilities.
`SHED` also provides translation from the python objects flowing through a `streamz` pipeline back into the Event Model enabling the use of data visualization and storage tools written for the Event Model. In addition to the translation features SHED passively tracks the provenance of produced data, capturing the data processing pipeline, data unique IDs, and the order of data insertion into the pipeline. The pipeline is captured in a way which is both human and machine readable, enabling searching and comparison of analyzed data sets. An extendable

*Corresponding author

system also captures information about the software environment in which the pipeline ran, including the conda environment, allowing in principle data analysis campaigns to be serialized into a database then later recovered and rerun, or adapted and then rerun and stored independently.

`SHED` is designed to provide a bridge between the data collected from high throughput experiments if they are, or can be put, into event model form (such as data from NSLS-II experiments), and the NSLS-II's data visualization and storage tools. This system has been used to write x-ray scattering data processing pipelines currently used in production at the NSLS-II XPD and PDF beamlines (28-ID-2 and 28-ID-1, respectively). The design and some applications of shed are described in greater detail in (Wright, 2020).

Figures

Example data processing pipeline with SHED nodes.

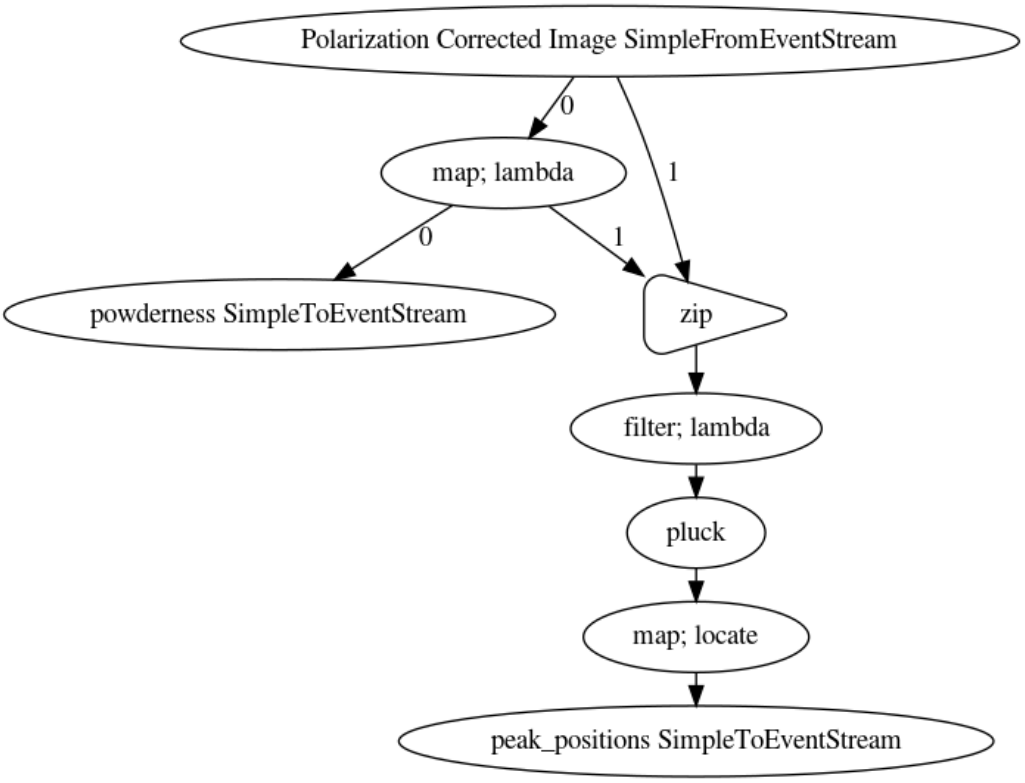

Figure 1: example pipeline

Acknowledgments

This work was supported as part of GENESIS: A Next Generation Synthesis Center, an Energy Frontier Research Center funded by the U.S. Department of Energy, Office of Science, Basic Energy Sciences under Award Number DE-SC0019212

References

Carbone, P., Katsifodimos, A., Ewen, S., Markl, V., Haridi, S., & Tzoumas, K. (2015). Apache flink: Stream and batch processing in a single engine. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, *36*(4).

Davison, A. P., Mattioni, M., & Samarkanov, D. (2014). Sumatra: A Toolkit for Reproducible Research. In *Implementing Reproducible Research*. https://doi.org/10.1201/b16868-9

Koerner, L. J., Caswell, T. A., Allan, D. B., & Campbell, S. I. (2020). A python instrument control and data acquisition suite for reproducible research. *IEEE Transactions on Instrumentation and Measurement*, *69*(4), 1698–1707. https://doi.org/10.1109/TIM.2019.2914711

Picard, R. (2018). *Hands-on reactive programming with python: Event-driven development unraveled with RxPY*. Packt Publishing Ltd.

Silva, C. T., Freire, J., & Callahan, S. P. (2007). Provenance for Visualizations: Reproducibility and Beyond. *Computing in Science Engineering*, *9*(5), 82–89. https://doi.org/10.1109/MCSE.2007.106

Wright, C. J. (2020). *Towards real time characterization of grain growth from the melt* [PhD thesis, Columbia University]. https://doi.org/10.7916/d8-5tdx-bb07