PROVENANCE ANALYSIS WITHIN A.N.T.A.R.E.S.

By

ERIC MICHAEL EVANS

_____

A Thesis Submitted to The Honors College

In Partial Fulfillment of the Bachelors degree
With Honors in

Computer Science

THE UNIVERSITY OF ARIZONA

M A Y 2 0 1 8

Approved by:

_____

Dr. Richard Snodgrass
Department of Computer Science

ABSTRACT

ANTARES, a software system for categorizing astronomic phenomena, employs a non-static pipeline. For scientific legitimacy, metadata needs to be recorded for each categorization. However, the pipeline processes up to fifty thousand alerts every thirty-nine seconds. As a result, there are limited resources for recording metadata in real-time. By allowing some overhead at startup, environmental metadata can be recorded, including what version of the pipeline was used. This recording can be achieved before processing has begun. Post-hoc analysis can be used to infer the rest of the needed metadata. Provenance, the computational history of the categorization of an alert, is entirely dependent on this metadata. DAFCAA, the Dynamic Alert Flow Computation Analyzer on ANTARES, is an interactive application used to assist in analyzing provenance post hoc. This application allows a user to view what configuration was used during a specified time period by retrieving source code from GitHub. By reading values queried from the database of alerts, the user can view the results of specific computations and therefore, alert classifications.

BACKGROUND

ANTARES is a distributed computing system, currently under development by the National Optical Astronomy Observatory in collaboration with the University of Arizona Computer Science Department, which will facilitate the categorization of astronomic phenomena gathered by the Large Synoptic Survey Telescope (LSST). These phenomena are referred to as *alerts*. ANTARES will process an estimated one to fifty thousand alerts per image every thirty-nine seconds. Throughout the ten-year lifetime of the LSST, the categorization of these alerts will lead to expanding knowledge of the Southern sky and will seek to find the rarest of the rare of astronomic phenomena.

INTRODUCTION

As ANTARES processes alerts through its pipeline, each alert is subject to a series of classification stages. In a stage, an alert can have one or more properties assigned to it, each paired with a calculated value. A stage downstream can then read a property value and in turn, use this to compute another property value. Thus, a property value may be dependent on a previously calculated property value. These values contribute to the alert eventually being classified as rare or ordinary (as most alerts are not rare). This inter-property dependence raises several questions regarding origin and legitimacy of calculation, which are fundamental for scientific research.

Provenance has been historically used to describe the lifetime of a painting and its components: the name of the artist, the date of completion, what museums the piece has been housed in, etc. In ANTARES, provenance is used analogously to refer to the results of computation and analysis that led to an alert being labeled or not labeled as rare. In addition to inter-property dependence, the pipeline is non-static: various stages can be altered, added or removed, and the order of these stages can change from night to night.

PROPERTY-BASED PROVENANCE

The process of analyzing the provenance of a property value can be separated into six broad questions:

1. *When was the property value created?* A timestamp is recorded for each image by the LSST and is stored at runtime. The position in the pipeline relative to other property value assignments can be inferred post hoc.
2. *What lines of the source code caused the creation of a property value?* This can be inferred by parsing stage code post hoc.
3. *How was the property value calculated?* This includes what inter-property dependencies existed in the source code and what external catalogs were read from. This can also be decided post hoc.
4. *What was the state of the machine that created the property value?* The cluster is standardized, so ANTARES will always operate in the same operating system environment on the same machines. Various state information is recorded to the database at each ANTARES runtime, to guarantee consistency.
5. *Which pipeline configuration was used at runtime?* This information, including the ordering of stages, is archived by GitHub, and the source code's SHA value is recorded at runtime.
6. *Why was the property value created?* This is dependent on the purpose of a stage, which is known well before runtime, by a vetting committee of astronomers.

POST-HOC PROVENANCE ANALYSIS

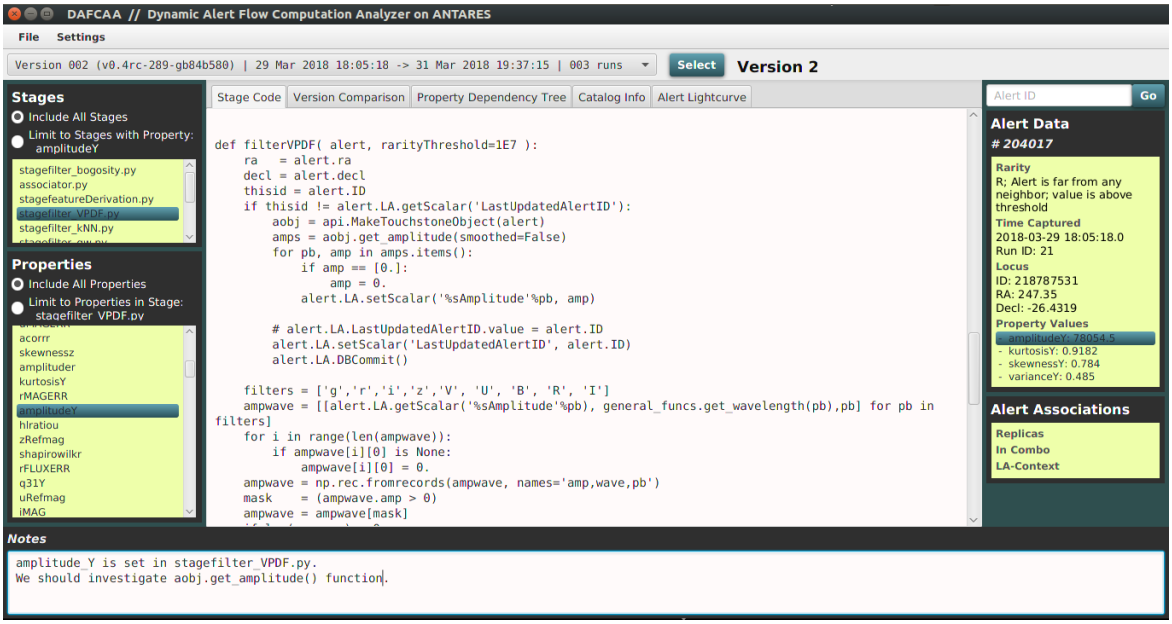
Because ANTARES is handling a vast amount of data at a high rate, there are little resources available to record the provenance for each alert to the necessary granularity. As a result, post-hoc analysis can be used to explain the computational history of any one alert. DAFCAA, the Dynamic Alert Flow Computation Analyzer on ANTARES, is an application that assists in answering the first five questions of provenance.

At DAFCAA startup, a user is presented with a list of source code *versions* to choose from. Each version represents a different pipeline configuration. This is defined by the SHA of the latest GitHub commit used at runtime. A single version may be used in multiple *runs*; that is, multiple nights of operation may pass before the pipeline configuration is changed. After the user selects a version, a list of stages and a list of properties are populated with those that were available during this time period. The user may also choose to query an alert to view information such as date of classification, rarity status, locus (location in the sky), and the values of its assigned properties.

In analyzing the provenance of a property value, it is necessary to know the stages that accessed this property – either to read or to set its value. This can be done by selecting a property from either the main list of properties or the list of property values for the queried alert. The user then has the option to limit the list of stages to include only those in which the selected property was used.

Conversely, the user may want to view what properties were accessed in a particular stage. Likewise, after selecting a stage, the user is given the option of limiting the main list of properties. When a stage is selected, the stage code is viewable in the center pane.

At any time during this analysis, the user has the option to record notes in the notes section, in the bottom pane. For documentation and record-keeping, the current state of DAFCAA can be exported to a YAML file. This file includes the user's notes, as well as the selected version, property, stage, and alert, along with an automated timestamp.



DAFCAA TECHNICAL DETAILS

Lines of Code

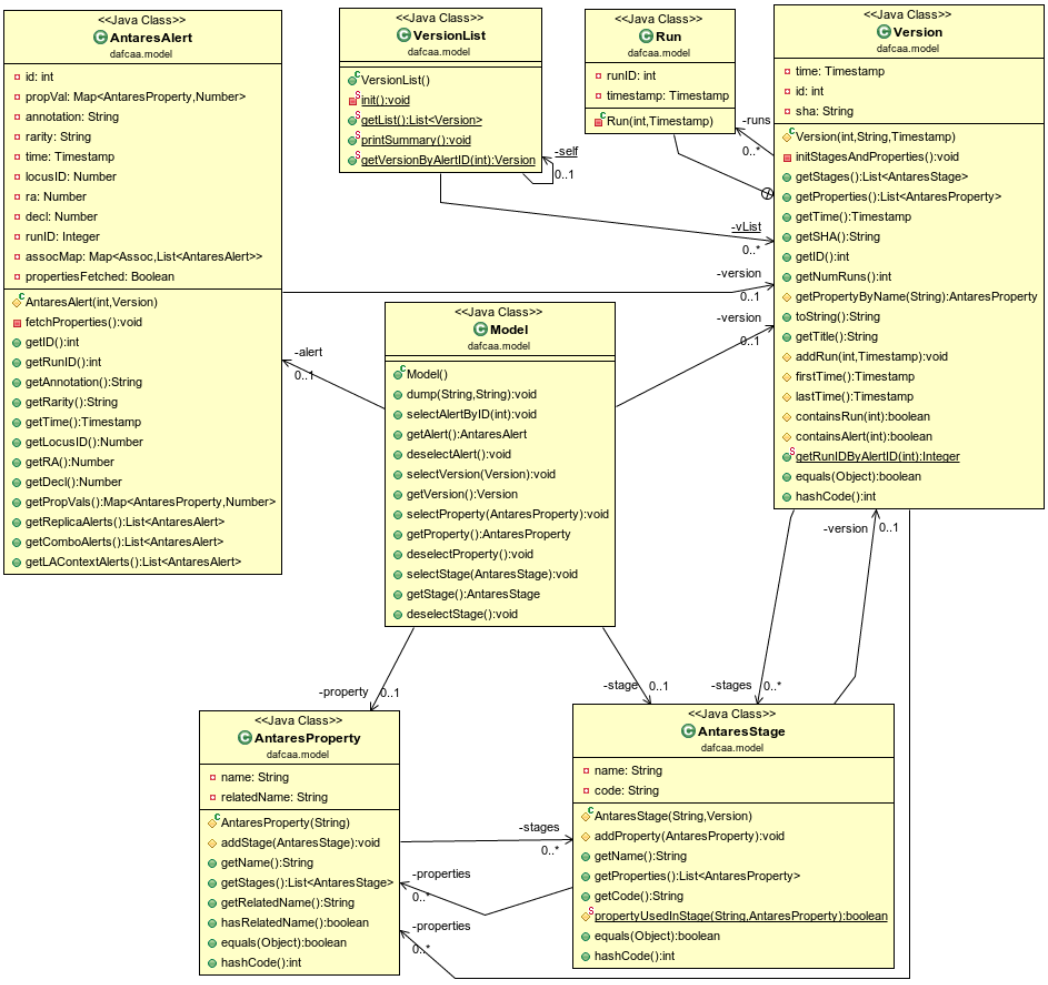
```
  36 config/Settings.java
 137 connections/GitHubConnection.java
  77 connections/LADBConnection.java
 649 control/Controller.java
  32 control/DialogMessage.java
  69 Main.java
  37 model/AlertOutOfDomainException.java
 201 model/AntaresAlert.java
 107 model/AntaresProperty.java
 107 model/AntaresStage.java
 165 model/Model.java
 272 model/Version.java
 131 model/VersionList.java
 383 view/view.fxml
 195 view/styles.css
2598 total
```

Directory Tree

```
.
├── dafcaa
│   ├── config
│   │   ├── README.md
│   │   └── Settings.java
│   ├── connections
│   │   ├── GitHubConnection.java
│   │   ├── LADBConnection.java
│   │   └── README.md
│   ├── control
│   │   ├── Controller.java
│   │   ├── DialogMessage.java
│   │   └── README.md
│   ├── Main.java
│   ├── model
│   │   ├── AlertOutOfDomainException.java
│   │   ├── AntaresAlert.java
│   │   ├── AntaresProperty.java
│   │   ├── AntaresStage.java
│   │   ├── Model.java
│   │   ├── README.md
│   │   ├── Version.java
│   │   └── VersionList.java
│   ├── README.md
│   └── view
│       ├── background.jpg
│       ├── README.md
│       ├── styles.css
│       └── view.fxml
├── Makefile
└── README.md

6 directories, 24 files
```

Model Class Diagram (dafcaa.model package)



FUTURE WORK

Moving forward, there are many more areas of alert provenance to be explored. It may be useful to analyze inter-alert associations, that is, alerts that depend on other alerts. Code slicing a stage for a property variable would be valuable to see the step-by-step provenance of a property value. This information could be used to generate a dependency graph of property values. DAFCAA could be extended to compare, side-by-side, the differences between multiple versions which would allow for quick, high-level analysis. The tool could display the differences in external resources, such as the state of the catalogs used for categorization.

CONCLUSION

During ANTARES runtime, there are limited resources for recording metadata, so post-hoc analysis is needed. DAFCAA is an interactive application used to assist in analyzing the provenance of a property value. This analysis is fundamental to answering questions regarding legitimacy of calculation need to confirm astronomic categorization.

ACKNOWLEDGMENTS

This work would not have been possible without the assistance of my thesis adviser, Dr. Richard Snodgrass. His constructive feedback, guidance, and enthusiasm throughout this year-long process have been valuable and are very much appreciated. My special thanks are also extended to those who helped me during my time at the National Optical Astronomy Observatory: Monika Soraisam, Gautham Narayan, Pete Wargo, Navdeep Singh, Zhenge Zhao, and Songzhe Zhu. I would also like to thank the University of Arizona Computer Science department and staff for supporting me throughout my research.