KeyW

Innovation Beyond Expectation

HOG

Unlocking Data for Analysts and Developers

7740 Milestone Parkway, Suite 150 | Hanover, Maryland 21076

KeyWcorp.com

TABLE OF CONTENTS

Introduction

Apache Pig (Pig) has had a presence in the Open Source community for the better part of a decade. In recent years, it has been a staple for programmers dealing with big data and cyber analytics as it is highly efficient in analyzing large data sets. Despite its popularity, however, there remains a high barrier to entry for analysts who wish to use it, yet have little-to-no coding abilities necessary for doing so.

KeyW recognized this need for an integrated development environment (IDE) where analysts with minimal - or no - coding experience can perform queries. Enter Hog - KeyW's IDE with a drag-and-drop interface for functional blocks, and automatic graphing and tabulation of the outputs from Apache Pig.

KeyW's Hog allows analysts to directly interact with large data sets without needing to fully understand Pig development or coding. By utilizing the functionality of Pig and displaying it in an easily comprehensible user interface, developers with little-to-no experience using Pig can use Hog to build analytic scripts that are comparable to those created by traditional Pig script developers.

Hog is a Bridge between Analysts and Developers

Because Hog uses the analysts' input to create Pig scripts, analysts can work with developers to further refine those scripts, improve efficiency, and so on. Analysts can also view the code directly if they wish to learn how to write Pig scripts themselves, while still being able to go back to the drag-and-drop interface when necessary.

By bridging this knowledge gap between analytic developers writing Pig scripts and analysts wanting to manipulate their data through Pig, the pairing of Pig and Hog results in less time and resources required to develop quality Pig scripts.

Hog is designed as a Node.JS powered website, so it can be deployed almost anywhere, and uses the locally installed instance of Pig.

Simple Page

The Simple Page is the centerpiece of Hog. From this page, users can select the functions (referred to as nodes) that they wish to apply to their data, and modify the script, all using a drag-and-drop interface. Hog provides a wide variety of nodes to choose from, including Relational Operators, Eval Functions, Load Store Functions, Load Types, Storage Types and Tuple Bag Map Functions. The Hog User Manual provides a complete list of nodes supported by Hog.

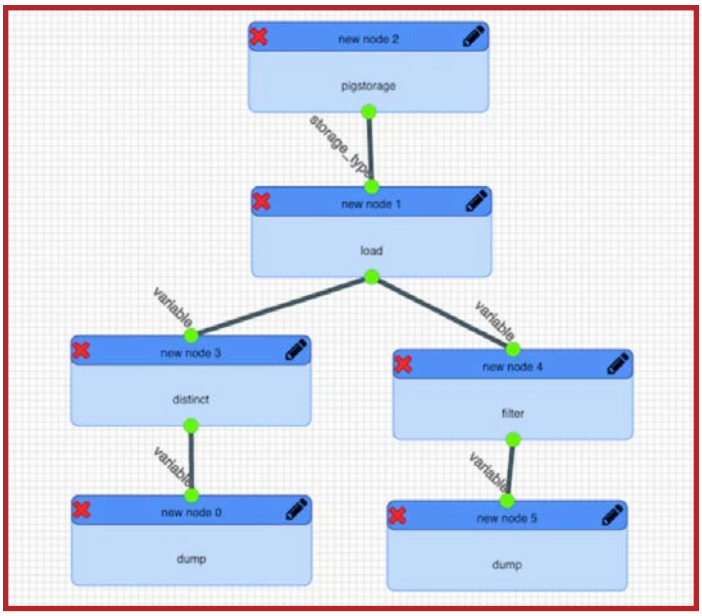

FIGURE 1 Hog loads data from a file selected by the user and passes it through two separate filters. It then generates the outputs, as notated in Figure 2.

After creating the Pig script, users can run it and view the outputs in various graph forms. Hog will classify the outputs into 5 categories: Results, Info Logs, Warnings, Errors, and Graphs, as shown in Figure 2. If the user chooses, they may simply select to show all outputs. In addition, the same settings that they used to create and view their outputs and scripts will be saved and carried over throughout their use of Hog.

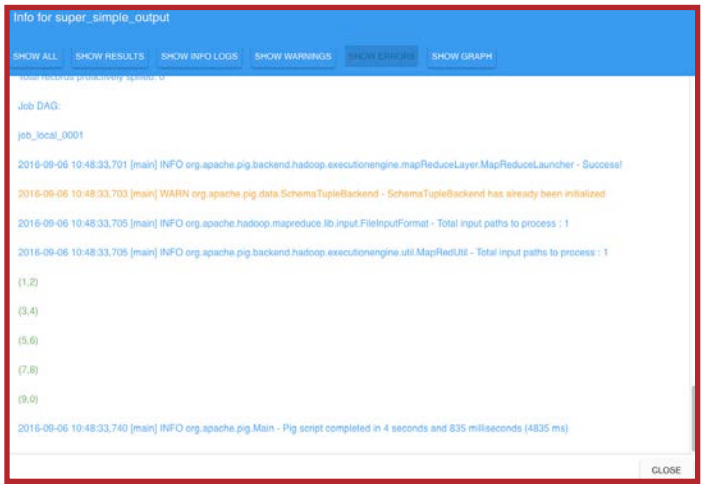

Figure 2 Displays the outputs of a script developed in either the Simple or Complex Pages. Each type of output is denoted in a unique color representing their classification.

For analysts who do not know how to create Pig scripts from scratch, this page will provide a simplified way of not only manipulating their data, but also sorting through various nodes and scripts via the List feature. List displays every script that the user has created in Hog and allows them to view, edit or delete as needed. This type of visualization is tailored for an analyst just beginning to learn Pig.

Complex Page

Though the Simple Page was developed around the idea of helping a novice Pig script writer, Hog also benefits the experienced Pig script developer. The Complex Page is where analytic developers can write Pig scripts in a traditional development environment while benefiting from Hog's features.

Features of the Complex Page include autocomplete functions and syntax colorization for usability. Also, unlike the Simple Page, users may upload and download Pig scripts to/from the Complex view.

Although the Complex Page is able to take scripts created in the Simple view and modify them, the same cannot be said for the Simple Page. If the user uploads or creates a script from the Complex Page, they currently cannot edit them in the Simple view.

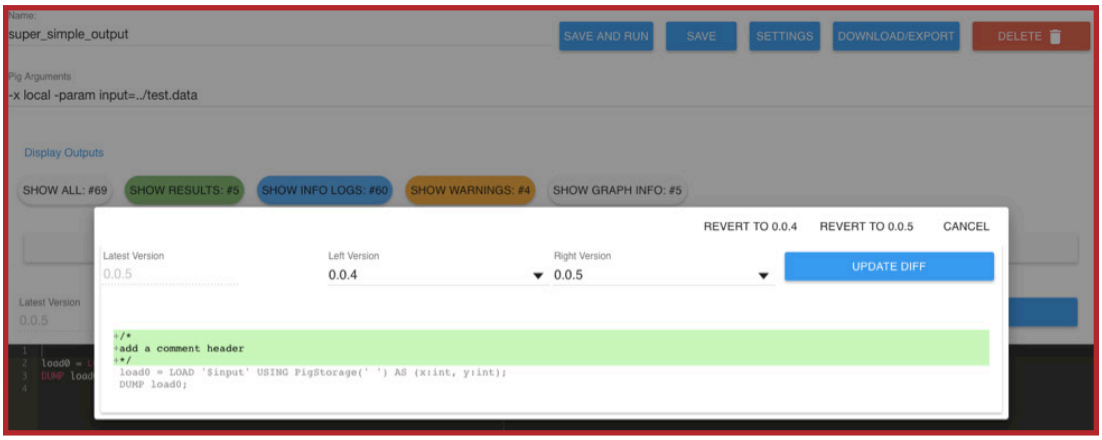

FIGURE 3 Hog's Versioning Tool

Editing through the Complex Page will also allow the developer to use the versioning tool. After an update to the script has been made, the user may decide to update the version of the script, which would allow them to return to that version of the script at any point in the future. This feature provides the developer with a way to review older editions of their code, a useful option for troubleshooting code.

Visualization of Results

Hog allows analysts and developers to run their scripts through the Hog interface, and to see the results in a variety of user-friendly formats. Results can always be viewed directly in tabular form, but the process of data analytics typically involves several phases of analysis, in which the results of one phase feed into the next, and the number of results gets successively smaller through the analysis.

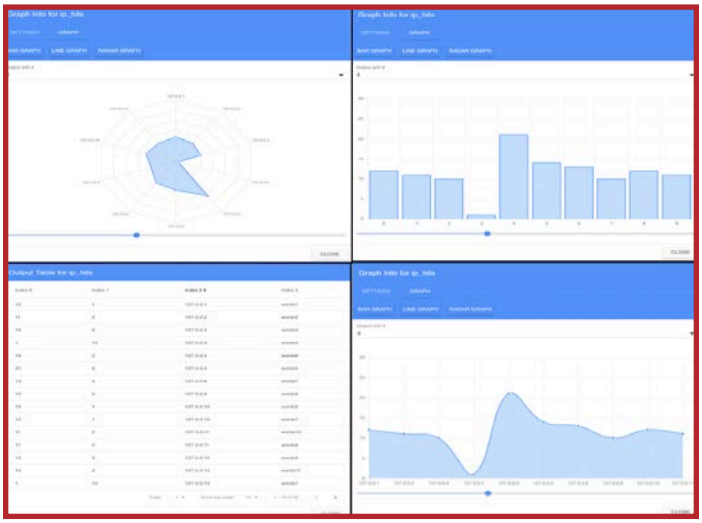

FIGURE 4 Graphed outputs generated by Hog

Using line graphs and other outputs can help analysts to spot the underlying distribution of data, as well as to find outliers - both of these tasks help in the development of the next phase of analysis, which can then be added to the script.

When script development and execution, and analysis of the results are all distinct operations using different tools, this complexity slows down the analyst. By bringing all of these aspects together, Hog improves the immediacy of the process, which speeds up development for all users.

Hog as a Learning Tool

Besides the creation of new Pig scripts, Hog enables amateur analysts to learn how to write Pig in the traditional way.

By toggling between the Simple view to the Complex view, users will be able to utilize the drag-and-drop functionality and then see how the generated code develops, allowing them to greater comprehend what it is that they are creating, and how it runs.

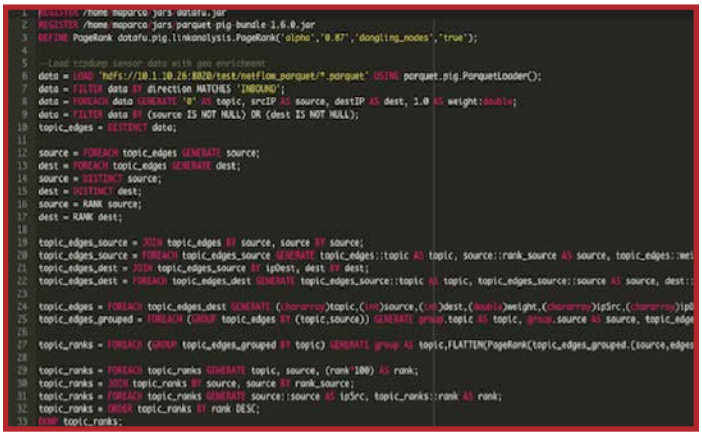

FIGURE 5 Complex view generated by Simple Page

This type of application of Hog would mean more developers with an understanding of Pig scripts. Eventually, users who began using the Simple page to graduate to the traditional development environment provided by the Complex Page.

Roadmap

Hog was conceived through a research and development project by KeyW's Cloud Computing Information Services (CCIS) division. A number of additional features may be added in future versions. These include:

- The ability to build a library of custom nodes, each representing a block of source code to be inserted into Pig scripts
- The ability to develop with the complex page and then see the simple page representation of that code
- Side-by-side view showing both Simple and Complex pages, with automatic update when the script in one of them is changed
- Integration with Git for more robust version control

Conclusion

Hog's capabilities benefit all levels of analysts and Pig script developers, from those who have expertise in the language to those who have never written a single script.

The Simple Page provides a shortcut for novice developers to learn Pig while still producing fully functioning scripts. The Complex Page provides a development environment for traditional Pig developers to work in, allows scripts made in the Simple view to be tweaked and helps novice developers garner a better understanding of Pig.

With the Hog application, Pig developers of all levels have a user friendly, end-to-end solution for writing all of their Pig scripts.

Authors

Connor Drew is a Technical Writer with CCIS IRaD and served as the Documentation Lead for the Hog project. He joined KEYW in the fall of 2015 after graduating from James Madison University in Harrisonburg, Virginia in 2014 with a Bachelor's in Writing, Rhetoric and Technical Communications.

Kevin Coxe is a Software Design Engineer in CCIS IRaD. He started at KEYW in October of 2015. Kevin is a full stack developer, mainly working with front-end javascript frameworks such as react and angular. He graduated from UMBC with his Bachelor's degree in Computer Science.

About KeyW

KeyW is a total solutions provider for the Intelligence Community's toughest challenges. We support the collection, processing, analysis and dissemination of information across the full life cycle of the Intelligence Community's mission. We employ and challenge the most talented professionals in the industry with solving such complex problems as preventing cyber threats, transforming data into intelligence and combating global terrorism. For more information, visit **KeyWcorp.com**.