A web interface for XALT log data analysis

Ruizhu Huang
Texas Advanced Computing Center
University of Texas at Austin
Austin, Texas
rhuang@tacc.utexas.edu

Weijia Xu
Texas Advanced Computing Center
University of Texas at Austin
Austin, Texas
xwj@tacc.utexas.edu

Robert McLay
Texas Advanced Computing Center
University of Texas at Austin
Austin, Texas
mclay@tacc.utexas.edu

ABSTRACT

XALT is a job-monitoring tool to collect accurate, detailed, and continuous job level and link-time data on all MPI jobs running on a computing cluster. Due to its usefulness and complementariness to other system logs, XALT has been deployed on Stampede at Texas Advanced Computing Center and other high performance computing resources around the world. The data collected by XALT can be extremely valuable to help resource providers understanding resources usages and identify patterns and insights for future improvements. However, the volume of data collected by XALT grows quickly over time on large system and presents challenges for access and analysis. In this paper, we describe development of a prototype tool to analyze and visualize XALT data. The application utilizes Spark for efficient data processing over large volume of log data and R for interactive visualization of the results over the web. The application provides an easy to use interface for users to conveniently share and communicate executable usage and patterns without prerequisite knowledge on big data technology. In this paper, we detail the features of this tool, current development status, performance evaluation and exemplar use cases.

Categories and Subject Descriptors

• Information systems~Data mining • Information systems~Web applications • Information systems~Users and interactive retrieval

Keywords

XALT, web interface, big data, log analysis, spark, R

1. INTRODUCTION

Historical usage data for HPC resources is collected and preserved for the purpose of audits, identifying usage patterns, understanding resource utilization and assisting future resource acquisition decisions. However the amount of the data collected is proportional to the size of the system, number of the metrics being gathered and growth over time. There are two challenges in re-use the log data. First the volume of the data makes even simple statistical analysis tasks not trivial. Secondly, as more diverse an complex data are gathering, the analysis calls for more complicated methods to facilitate understanding both raw data and analytical results. To address those two challenges, we developed a prototype tool to utilize Spark [19, 20] for efficient data processing and analysis and R [9] for providing interactive visualization over the web.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions@acm.org.
XSEDE16, July 17-21, 2016, ,
© 2016 ACM. ISBN 978-1-4503-4755-6/16/07…$15.00
DOI: http://dx.doi.org/10.1145/2949550.2949560

Our project is motivated by analysis and curation needs of XALT data gathered on Stampede [4]. XALT project is designed to track linkage and execution information for applications that are compiled and executed on any Linux cluster, workstation, or high-end supercomputer. In addition, XALT will also monitor and track individual code execution, and in turn will alert support staff and/or users regarding the basic causes of problems preventing their jobs from running properly. XALT can also collect metrics intended to improve training, documentation, management and outreach programs [2]. The data collection is transparent to the users. The data collected are stored in the database that can be mined to generate a picture of the compilers, libraries, and other software that users need to run their jobs successfully, highlighting the products that our researchers do and do not use [1]. Because of its usefulness and complementariness to existing logs and databases, XALT has been in production for over two years and in use at many HPC centers around the world. Centers with XALT have a much better understanding of library and executable usage and patterns [2].

However, as more detailed information is collected over time, the volume of the data becomes increasingly hard to be accessed, analyzed and curated by the researchers. For example, one month of partial log data on Stampede can result a JSON file as big as 3GB with thousands of records each with tens of attributes. The analysis of the log data requires scalable data processing methods to ensure efficiency of the data processing. Additionally, resource providers, data curators and end users have different interests and requirements for re-using the same data. Therefore the data analysis process also needs to be flexible for different user's needs and to be interactive for enabling easy exploration of the data online.

Our application development combines the efficiency of the large-scale data processing with web-based user interface and visualization to meet those requirements. At the core of this application is an analysis framework implanted using Spark as foundations to scale data processing analysis efficiently. Our application currently supports a set of statistical and association analysis. With web-based applications, users are able to access big data analysis via a uniform and user-friendly environment, the web browser. A web interface for big data analysis does not require extensive training of shell commands and a lengthy learning curve to get familiar with Spark framework, specific programming languages and big data applications. Most importantly, a web interface provides an efficient way to explore big data sets dynamically with adjustable runtime parameters.

The goal of this application is to provide a way to share institutional knowledge and big data analytics among users, software developers, and support staff. In Section 2, we present background and related work to our project. Section 3 describes our methods and implementation details. In Section 4 we present performance evaluations of the data processing using log data generated on Stampede from 2014 to 2015 to show the scalability of the application. In Section 5, we present details of our web interface together with exemplar use cases to demonstrate potential values of this tool. We conclude and discuss ongoing works in Section 6.

2. BACKGROUND AND RELATED WORK

Log analysis usually requires aggregation for statistical inference. Most previous log analysis focus on computer system by using machine-learning techniques, especially anomaly detection to discover interesting log messages. The system-level predictive models help automate or provide insights for resource provisioning, capacity planning, workload management, scheduling, and configuration optimization [11, 13, 14]. And the user-level log analytics in this paper provides useful information on usage patterns by science field, applications, core, etc.

The data collected by XALT have been analyzed in [1, 2]. The usage reports including compiler usage, top used libraries, top executables and software pruning have been extracted from XALT database at the National Institute of Computational Sciences (NICS) on a Cray XC30 supercomputer called Darter. Usage reports for three other supercomputing centers including Texas Advanced Computing Center (TACC), Research Computing Center (RCC) at Florida State University and King Abdullah University of Science and Technology (KAUST) are also provided for comparison [2]. In those papers, the usage of link programs aggregated using SQL queries shows different patterns across computing centers through pie chart comparison and top used libraries grouped by module name in NICS are demonstrated on bar plots.

Our application currently supports distribution analysis, association analysis and sequential pattern mining with many options for users to interactively adjust, explore and visualize large volume of log data with respect to time series pattern, applications pattern, science domain pattern, parallelism pattern, etc.

To meet the generic analysis needs and increasing volume of log data, we utilize Spark for efficient large-scale data processing. Spark is an open-source cluster computing framework originally developed by AMPLab at UC Berkeley [21]. One of the big advantages of Spark is that Spark runs in-memory on the cluster. And Spark does not tie to Hadoop's MapReduce two-stage paradigm. This makes repeated access to the same data much faster [8].

The results from the Spark analysis are then used for visualization and interactive online access with implementation using R [9]. R is an open source programming environment that supports a large variety of common statistical analysis and data analysis tasks. Due to its high extensibility and continuous developments by its active user community, R has become a popular analytic environment by many researchers across scientific fields over the past decades [18]. One of the most popular packages, Shiny is an R package for building interactive web applications [16]. It integrates data analyses into interactive web applications and consists of two important components. One component is a user-interface script controlling the layout and appearance of the app. The other component contains the instructions to support interactions of the app. This paper provides a big data analytic solution for XALT data and a user-friendly web interface for stakeholders. Static analysis techniques are usually limited by the size and complexity of the target system. Log volume can be excessive in a large system over time, e.g. Stampede at TACC generates around 2GB log data per month, so log analytics requires an ability to ingest a large amount of information and then boil that information down into a more consumable data set that summarizes the important bits from the interactions. In addition to the statistical summary of the data, our application also supports association analysis to infer correlations among different types of data gathered from the XALT data. The web interface provides a uniform media for stakeholders to explore, communicate and compare various aspects of usage information among supercomputing centers.

3. METHODS DESIGN AND IMPLEMENTATION

3.1 Data Processing Overview

The Figure 1 shows the overall data processing workflow. We use data gathered from Stampede cluster. The raw data is provided by XALT project in JSON format and organized on monthly basis. All user identifiable information, such as user names, has been removed or replaced in the input data sets. The data has 18 columns and about seven million rows of records for the one-year logs. User related information includes *'user'* – anonymized unique user id for the account owner, *'allocation'* – the administrative account users run jobs against and *'field_of_science'* – the associated scientific domain of the user. Job related information includes *'exec_path'* – the name of the real executable in a job submission, *'num_cores'* – the number processors used during a job, *'run_time'*, etc. More details of the data fields are listed in the data dictionary [3].

The JSON files can be stored in either local file system or in a Hadoop Distributed File System (HDFS) [17]. Reading from HDFS is more efficient than from the local file system when running Spark analysis on top of the same Hadoop clusters. During the data processing steps, the input data is cleaned and filtered including removing records with missing data, converting the data fields into appropriate types, and normalizing the data values.

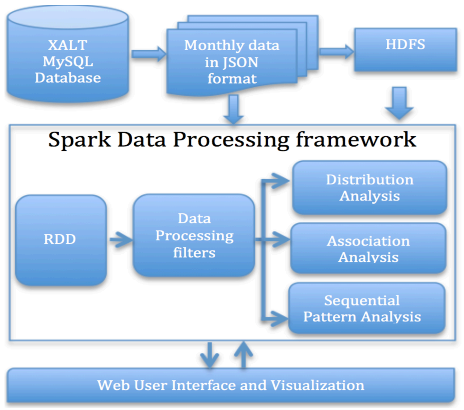

XALT MySQL Database
Monthly data in JSON format
HDFS
Spark Data Processing framework
RDD
Data Processing filters
Distribution Analysis
Association Analysis
Sequential Pattern Analysis
Web User Interface and Visualization

Figure 1. Overall data processing and analysis workflow

After the data is ingested into Spark cluster, filters can be used to select subset of data based on specific values in a given data field. The pre-defined filters include filter based on the origins of the executable, number of occurrences and dates. The implementation utilizes Structured Query Language (SQL) through which additional more complex filters can be easily added upon future use cases. In our testing, we implemented a community-code-filter that filters out jobs running user's private code. This filter allows the researchers just focusing on analysis of known community software and their usages. The analysis results from Spark are passed directly into Shiny web front for presentation. The web user interface also enables users to run analysis or visualize the analysis results interactively.

3.2 Application Features and Supports

There are three analysis features currently supported. Data distribution analysis allows users to select two data fields in the XALT data to see how one related to the other. Association analysis enables user to investigate associations among values in the subset of data fields defined by users. The sequential pattern analysis enables users to see if there is any sequential pattern over subset of fields over the time.

3.2.1 Data Distribution Analysis

Given two data fields, the distribution analysis carries out as the following. Unique values from each data field are first identified. The unique values are used as labels to construct the distribution matrix between the two fields. The occurrences of all the unique value combinations between two fields are aggregated and used to populate the distribution matrix. The distribution analysis processes will result an m x n matrix where m and n are the number of unique values in each field respectively.

For example, the distribution analysis can be used to show how frequently different scientific fields using community codes. To do so, a user can select *'exec_path'* and *'field_of_science'* as two input data fields. In this example, the values in *'field_of_science'* can also be standardized and reduced according to common conventions. The result distribution matrix can be used by the web interface to generate histograms with a color map to visualize values in the matrix. The web interface provides an option for user to choose how many top applications to be shown on the histogram.

3.2.2 Data Association Analysis and Sequential Pattern Mining

The data association analysis generates inferences between values from two or more fields of the data in a given condition using FP-Growth algorithm [7].The analysis starts with selecting and aggregating subset of data specified by the input parameters as a list of records, also known as *transactions*. The analysis algorithm will scan the selected data set to compute the frequency of each value, also referred as an *item*, and store the frequency value and co-occurrence with other *items, collectively referred as itemset,* in a tree structure, named frequent pattern tree (FP-tree). Then the frequent item sets can be identified from the FP-tree to generate inferences among subset of values.

In the example described in the previous section, the distribution analysis result shows an intuitive view of how executable is used across different science fields. But it doesn't directly show potential usage patterns between specific executable used by a user and the scientific field identified by the same user. To answer the latter question, the association analysis can be conducted between the *'exec_path'* and *'field_of_science'*. In this example, the analysis will first aggregate *field_of_science* data and *exec_path* associated with each user. The association analysis will then generate a list of inference rules to indicate the likelihood of a subset of executable, Y, to be used by a user from a field_*of*_science X.

The FP-Growth algorithm can be adapted for sequential pattern mining to identify sequential usage patterns [7] To support sequential pattern mining, data are first sorted and then aggregated based on the job submission time per user. The association analysis can generate a sequence of executable likely to be used in order by the user. The result of the sequential pattern mining is a list of ordered combinations of values.

Both association analysis and sequential pattern analysis are implemented with MLlib library [12] in Spark. In addition to specify the fields of data to be used for the analysis, additional parameters are required for that analysis as well. Common to both algorithms, one of the most important parameters is the *"minimum support value"*, $supp(x)$. $supp(x)$ stands for the support value of an *itemset* x and is defined as the percentage of transactions in the data set which contain the *itemset*. This value is used to select patterns that are used to build the FP-tree. Another important parameter is the confidence value $conf(X \Rightarrow Y)$, which is used to indicate the likelihood of the inference between itemset X and itemset Y. The $conf(X \Rightarrow Y)$ is defined as

$$\text{conf}(X \Rightarrow Y) = supp(X \cup Y)/supp(X)$$

An association rule is defined as $X \Rightarrow Y$ which satisfies: $supp\ (X \cup Y) \geq \sigma$ and $conf(X \Rightarrow Y) \geq \delta$ where σ and δ are the minimum support and minimum confidence, respectively.

There are other parameters that can be specified. Those values can affect the performance and the result analysis. We refer interested readers to the original algorithm papers for more details [7, 15]. Among our existing use cases, we do not identify significant sequential patterns. Therefore, the rest of the papers are focused on the association analysis. Another important measure for association analysis is used throughout this paper is *lift*. The *lift* of a rule is defined as

$$lift(X \Rightarrow Y) = supp(X \cup Y)/(supp(X)supp(Y)).$$

It can be interpreted as the deviation of the support of the whole rule from the support expected under independence given the supports of both sides of the rule. Greater lift values ($\gg 1$) indicate stronger associations. Measures like support, confidence and lift are generally called *interest* measures because they help with focusing on potentially more interesting rules [6].

3.2.3 Visualization and Web Interface

The web user interface is vital to enable users to conduct analysis interactively online and to visualize the results. The Shiny interface we developed allows users to intuitively choose or change time range and various parameters for analysis and plotting. First, users can select a certain time range to examine the distribution and association pattern of various variables. Second, for the distribution analysis users can choose which column to be groups on *x* axis and which column to be categories on the legend. The results will be shown as a stacked bar chart. The interface also allows the user to choose how many top group-by columns as bars on the distribution bar chart. Third, the interface provides flexibility of changing the confidence and support threshold to filter out weak association *itemsets*. Similar to distribution analysis, a group-by column can be chosen and columns to be aggregated can be selected on the web interface. How many top rules to be displayed is also adjustable. Finally, the interface supports setting various options of the visual presentations. The details of the web interface and examples are provided in Section 5.

4. PERFORMANCE RESULTS

The data set used in this paper is provided by the XALT project. The data includes job submission records from July 2014 to June 2015. The data is exported from the database as twelve individual files in JSON format. The size of the monthly files varies greatly from several megabytes to almost 3 GB. We used this data set to test the performance and efficiency of the data processing routine. A synthetic data set (70 million records, 130GB) was generated using the one-year XALT log data for the scalability test. All of our testing were done using a Hadoop cluster dynamically provisioned by Wrangler [10]. In this Hadoop cluster, two data nodes are available to for Spark processing jobs. Each node has two Intel Haswell CPUs with 12 processing cores each, 128 GB of DDR4 memory, and allocated with 4 TB of flash storage used for building Hadoop distributed file system.

4.1 Data Analysis Complexity

Figure 2 shows the performance of running time in seconds when using different size of the input data. The plot was generated by running monthly data, quarterly data and yearly data. The blue line shows the time for running data association analysis and the red line show the time of running distribution analysis. In both cases, the running time is proportional to the number of records. The results show that the running time is linear to the number of records in the input data. In addition, a synthetic data set (70 million records, 130GB),10 times the one year dataset, was generated to test the scalability of the analysis on Figure 3. The runtime for distribution and association analysis is 64 and 73 seconds respectively running on a cluster with 32 compute nodes.

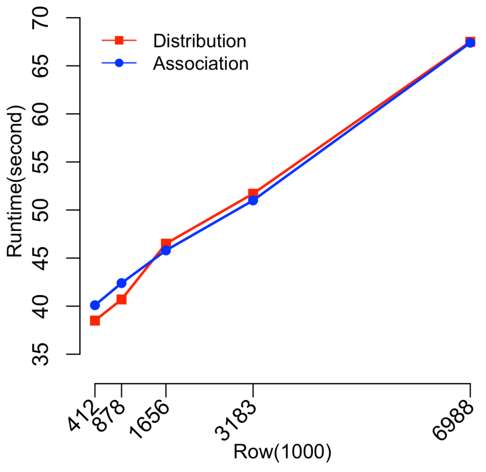


Figure 2 Running time of data analysis over number of records processed.

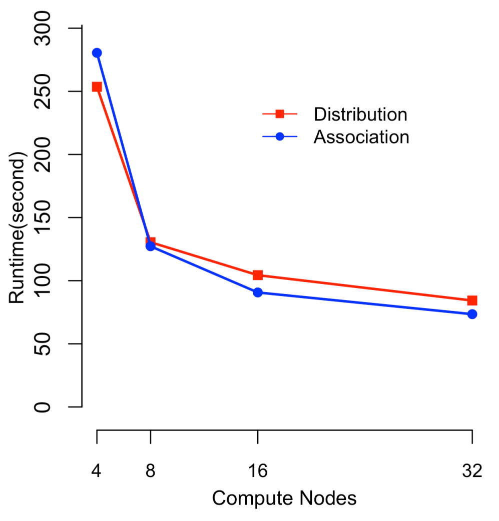


Figure 3 Running time of data analysis on 130GB synthetic data over number of compute nodes

4.2 Data Analysis Scalability Using Parallel Execution.

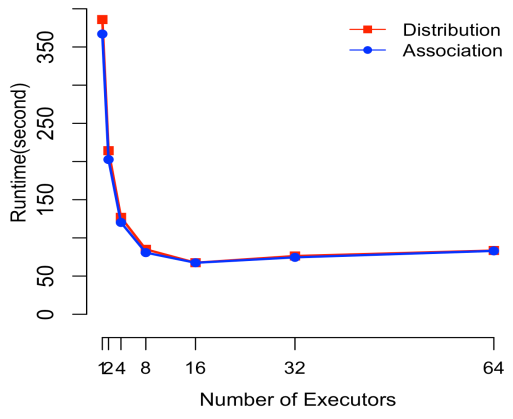
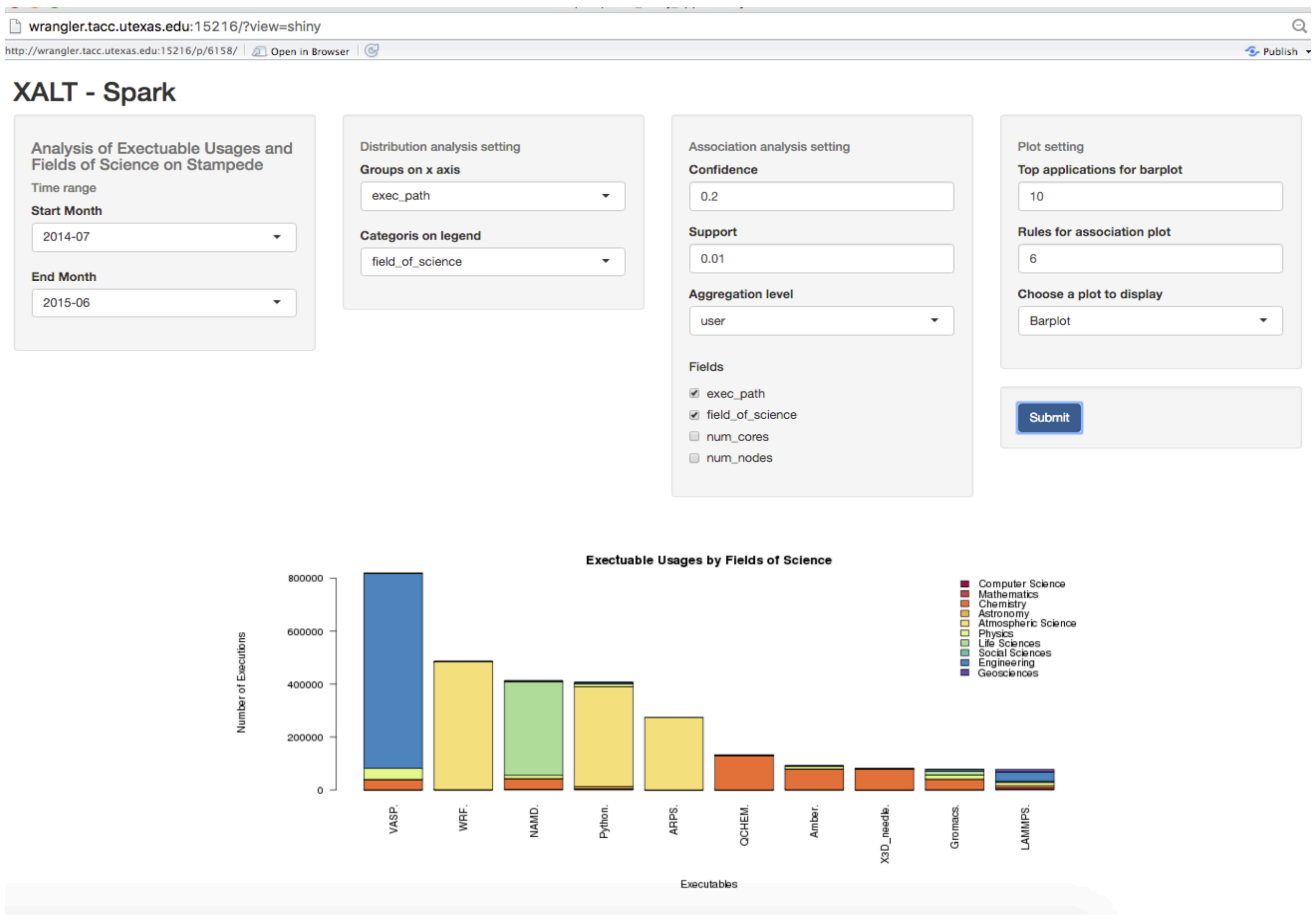


Figure 4 Running time (seconds) vs. number of executors used to conduct full set of analysis.

Figure 4 shows the analysis performance scales with the increasing computing resources used for the computation. With Spark processing framework, the degree of the parallelism is determined by the *number of executor* and *number of cores used per executor*. Increasing either the number of executors or number of cores used per executor can dramatically decrease the runtime. However the efficiency is also depends on the cluster resource (i.e. numbers of cores and memory) and the computation itself. For our testing using a three-node Hadoop cluster. The performance evaluation was conducted using one-year data which has 7 million records and 12.8GB in total size. The performance with various setting of number of executor and number of cores per executor was tested and the optimal setting is to use 16 executors each with three cores per executors. As shown in Figure 3, the running time is reduced significantly from using one executor (three total parallel processes) to four executors (twelve total parallel processes). With more executors, the performance benefits diminish gradually. When using more than 16 executors (48 total parallel processes) executors, the performance is no longer increases but slightly decreased.

5. Web User Interface and Use Case Examples

The web user interface is developed using R with the Shiny package. A reactive expression is used to prevent unnecessary rerunning of the analysis program if the parameters of time range, distribution setting or association setting remain unchanged. Therefore, changing settings of plotting options will not require running the analysis program again. The server.R uses a system call function to run the Spark Analysis program and create a distribution matrix and a association rule model. The results of the association analysis are stored in Predictive Model Markup Language (PMML) format [5] and used by the web interface for visualization. The visualization uses another R package, *AruleVis*, together with *Shiny*. Figure 5 shows a screenshot of the web user interface. There are four groups of user controls, data selection control, distribution analysis control, data association analysis control and visualization graph controls. The first group allows users to select start and end month to define the time range of the analysis. If the user wants a specific month, the user can select the same month as both start month and end month. The second group is to define which fields are used for the distribution analysis and how the results will be visualized. The group includes two drop-down boxes. One is for specific field to be used as label for each bar. And the other is used to specify the field to be used for different categories. Different categories will be mapped to a color map. The mappings are displayed in graph legend on the top right of the plot. The third group exposes settings for association analysis. The controls allow users to specify how the records should be aggregated through the "aggregation level" selection field and values of confidence level and support level to be used in association analysis algorithm.


wrangler.tacc.utexas.edu:15216/?view=shiny

http://wrangler.tacc.utexas.edu:15216/p/6158/

XALT - Spark

Analysis of Exectuable Usages and Fields of Science on Stampede

Time range

Start Month: 2014-07

End Month: 2015-06

Distribution analysis setting

Groups on x axis: exec_path

Categoris on legend: field_of_science

Association analysis setting

Confidence: 0.2

Support: 0.01

Aggregation level: user

Fields

- ✓ exec_path
- ✓ field_of_science
- ☐ num_cores
- ☐ num_nodes

Plot setting

Top applications for barplot: 10

Rules for association plot: 6

Choose a plot to display: Barplot

Figure 5 Screenshot of Shiny web interface

The prototype application has been used to help data curators to assess the value of the XALT data and how to reuse them [4]. Here we show a few other possible use case examples on what the application can help researchers easily achieve.

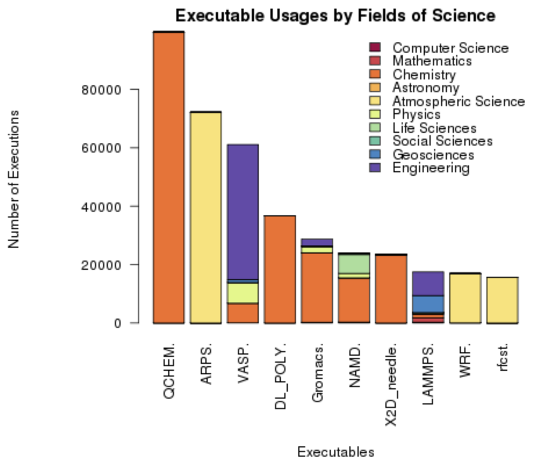


Figure 6 Distribution analysis of 3rd quarter in 2014

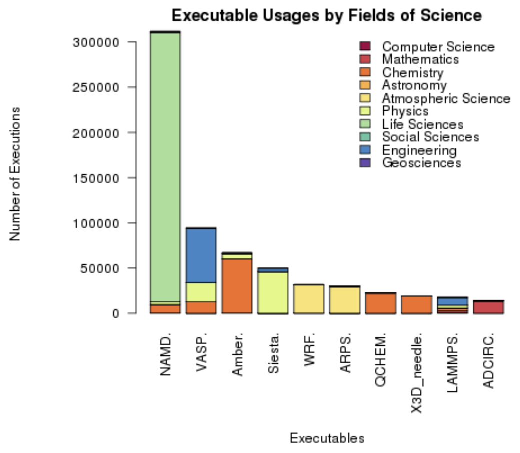


Figure 7 Distribution analysis of 4th quarter in 2014

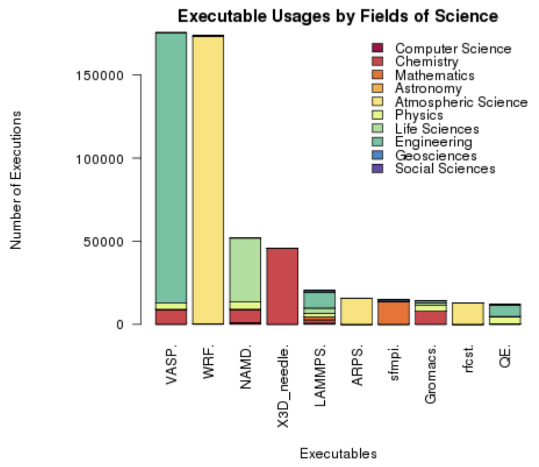


Figure 8 Distribution analysis of 1st quarter in 2015

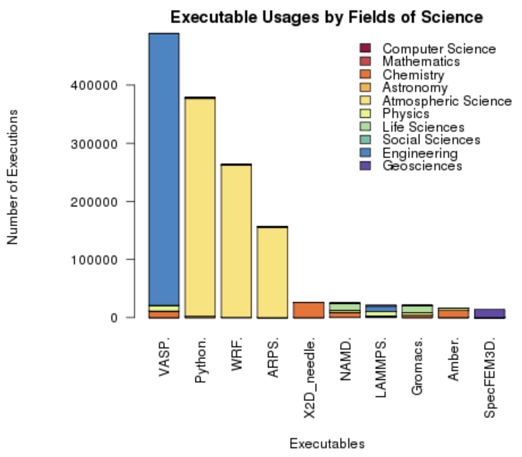


Figure 9 Distribution analysis of 2nd quarter in 2015

Figure 6 to Figure 9 shows quarterly comparison of distribution analysis of the data from the third quarter of 2014 to second quarter in 2015. The histogram shows significant differences in top applications as well as top fields of science of using those tools.

For comparisons, Figure 10 and Figure 11 generated by switching the selections in the distribution setting on the web interface show the distribution analysis over the entire year by executable and fields of science. We can see the most used executable is VASP, while atmospheric science users have most number of executions by using three different executables.[1] In addition, we also can tell some executables are not exclusively used by only one specific field of science in Figure 9 and vice versa in Figure 10.

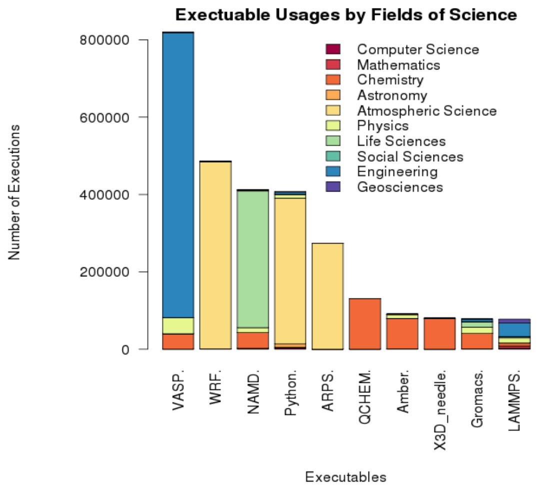


Figure 10 Executable by field of science, one year

[1]Note that current data paints an incomplete picture. Stampede XALT and accounting data (2015Q4) shows that non-MPI activity represents 48% of total job count.

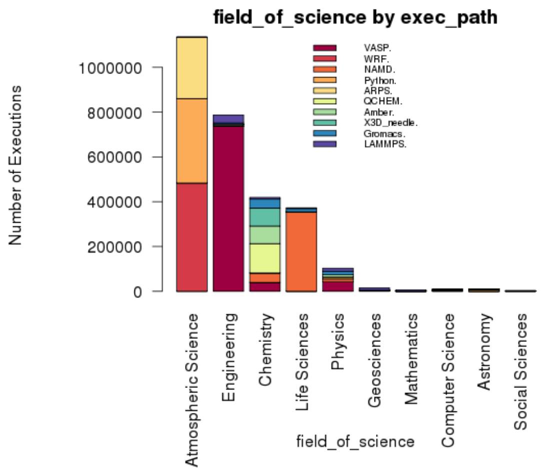


Figure 11 Frequency of field of science per executable

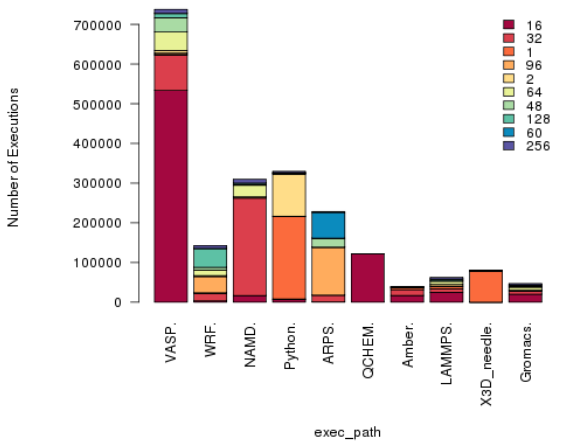


Figure 12 Frequency of executable per cores

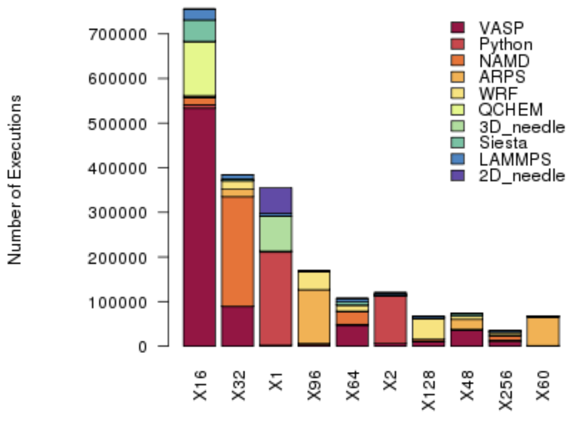


Figure 13 Frequency of number of cores per executable

Similarly, Figure 12 and Figure 13 show the distribution by executable and number of cores used. From these two figures we can tell core usage pattern by each executable and provide future suggestions on how many cores are commonly used by a specific executable. For example, we can see most VASP executions are requesting 16 cores.

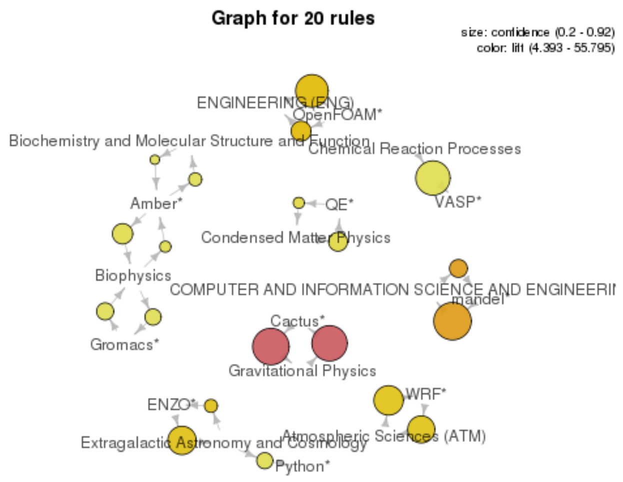

Graph for 20 rules
size: confidence (0.2 - 0.92)
color: lift (4.393 - 55.795)
ENGINEERING (ENG)
OpenFOAM*
Biochemistry and Molecular Structure and Function
Chemical Reaction Processes
Amber*
QE*
VASP*
Condensed Matter Physics
Biophysics
COMPUTER AND INFORMATION SCIENCE AND ENGINEERI
mandel*
Cactus*
Gromacs*
Gravitational Physics
ENZO*
WRF*
Extragalactic Astronomy and Cosmology
Atmospheric Sciences (ATM)
Python*

Figure 14 Top 20 rules by *Lift* in one year

Figure 14 shows the top 20 inferences rules between field of science and the community code used by that field of science. In this visualization, each inference rule is visualized as a directed graph. The nodes are text labels. The direction of the arrow from one text label to the other indicates an inferred association rule between the two labels. The colored circle in the path indicates the numerical measures of the inferences rules. The size of the circle is used to indicate the *confidence* value and the color is used to represent the *Lift* value. For example, a user from *"gravitational physics"* is likely to use the *"cactus"*. As indicated by the size of the circle, this inference is symmetry that is a user using "cactus" also is likely from the *"gravitational physics*". However, the other two pairs of association appear to be more directional. i.e. A user using *"mandel"* is more likely from the "computer science". But a user from *"computer science"* may not use *"mandel"* exclusively. The users from *"Engineering"* seems more likely to use *"OpenFoam"*, but *"OpenFoam"* are not used exclusively by the "Engineering field". We also conducted similar analysis on quarterly data and discovered a few additional inferences of interests. For example, from data of first quarter in 2015, we derived inference rules between

Condensed matter physics ➔ QE(QUANTUM ESPRESSO)

However, from data of second quarter in 2015, the strongest inference rule related to Condensed matter physics become:

NRLmol(Molecular Orbital Library) → Condensed matter physics

Additional similar changes are also noted. The strongest association for Atmosphere Science changed from ARPS to WRF between the third and fourth quarterly data in 2014.

6. CONCLUSIONS AND ONGOING WORK

In this paper, we present our preliminary work in developing a web-based application that enables interactive analysis with XALT data and supports online visualization of the results. This tool is the first of its kind to enable interactive scalable statistical and association analysis over the XALT data. The goal of this tool is to be both efficient for large data and flexible to meet different analysis needs. We demonstrated the scalable performance of the application with one-year XALT data from Stampede cluster. The results show the analysis performance can be scaled with increasing computing resources. The tools have been used for data curation and re-use [1]. Additional use cases are shown here to demonstrate the flexibility of the tool to meet different analysis requirement.

Our ongoing works include improving its user interface to add more options for user to select data and choose analysis process. We are also working on to extend the application with additional data source such as real time monitoring data, system utilization data and allocation data in order to provided a more comprehensive view of the resource utilization and geographic distribution patterns of users.

Acknowledgement

This project has been funded and supported on Wrangler cluster, which is generously funded by the National Science Foundation (NSF) through award ACI-1447307.

Reference

[1] Agrawal, K., Fahey, M.R., McLay, R. and James, D. 2014. User environment tracking and problem detection with XALT. *Proceedings of the First International Workshop on HPC User Support Tools* (2014), 32–40.

[2] Budiardja, R., Fahey, M., McLay, R., Don, P.M., Hadri, B. and James, D. 2015. Community use of XALT in its first year in production. *Proceedings of the Second International Workshop on HPC User Support Tools* (2015), 4.

[3] Data Dictionary: xalt_run table: 2015. *http://web.corral.tacc.utexas.edu/XALT/DataDict.pdf*. Accessed: 2016-04-28.

[4] Esteva, M., Sweat, S., McLay, R., Xu, W. and Kulasekaran, S. 2016. Data Curation with a Focus on Reuse. *proceedings of Joint Conference on Digital Library 2016 (JCDL'16)* (Newark, NJ, USA, 2016).

[5] Guazzelli, A., Zeller, M., Lin, W.-C. and Williams, G. 2009. PMML: An open standard for sharing models. *The R Journal*. 1, 1 (2009), 60–65.

[6] Hahsler, M. and Chelluboina, S. 2011. Visualizing association rules: Introduction to the R-extension package arulesViz. *R project module*. (2011), 223–238.

[7] Han, J., Pei, J. and Yin, Y. 2000. Mining frequent patterns without candidate generation. *ACM Sigmod Record* (2000), 1–12.

[8] Huang, R. and Xu, W. 2015. Performance Evaluation of Enabling Logistic Regression for Big Data with R. *2015 IEEE International Conference on Big Data*. (2015).

[9] Ihaka, R. and Gentleman, R. 1996. R: a language for data analysis and graphics. *Journal of computational and graphical statistics*. 5, 3 (1996), 299–314.

[10] Jordan, C., Walling, D., Xu, W., Mock, S.A., Gaffney, N. and Stanzione, D. 2015. Wrangler's user environment: A software framework for management of data-intensive computing system. *Big Data (Big Data), 2015 IEEE International Conference on* (2015), 2479–2486.

[11] Martino, C. Di, Cinque, M. and Cotroneo, D. 2012. Assessing time coalescence techniques for the analysis of supercomputer logs. *Dependable Systems and Networks (DSN), 2012 42nd Annual IEEE/IFIP International Conference on* (2012), 1–12.

[12] Meng, X., Bradley, J., Yavuz, B., Sparks, E., Venkataraman, S., Liu, D., Freeman, J., Tsai, D.B., Amde, M. and Owen, S. 2015. Mllib: Machine learning in apache spark. *arXiv preprint arXiv:1505.06807*. (2015).

[13] Oliner, A., Ganapathi, A. and Xu, W. 2012. Advances and challenges in log analysis. *Communications of the ACM*. 55, 2 (2012), 55–61.

[14] Pecchia, A., Cotroneo, D., Kalbarczyk, Z. and Iyer, R.K. 2011. Improving log-based field failure data analysis of multi-node computing systems. *Dependable Systems & Networks (DSN), 2011 IEEE/IFIP 41st International Conference on* (2011), 97–108.

[15] Pei, J., Han, J., Mortazavi-Asl, B., Wang, J., Pinto, H., Chen, Q., Dayal, U. and Hsu, M.-C. 2004. Mining sequential patterns by pattern-growth: The prefixspan approach. *Knowledge and Data Engineering, IEEE Transactions on*. 16, 11 (2004), 1424–1440.

[16] RStudio, I. 2013. shiny: web application framework for R. *R package*. (2013).

[17] Shvachko, K., Kuang, H., Radia, S. and Chansler, R. 2010. The hadoop distributed file system. *Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on* (2010), 1–10.

[18] Xu, W., Huang, R., Zhang, H., El-Khamra, Y. and Walling, D. 2016. Empowering R with High Performance Computing Resources for Big Data Analytics. *Conquering Big Data Using High Performance Computing*. Springer.

[19] Zaharia, M. 2011. Spark: In-Memory Cluster Computing for Iterative and Interactive Applications. *Invited Talk. NIPS Big Learning Workshop: Algorithms, Systems, and Tools for Learning at Scale (Granada, Spain. December 12--17, 2011)* (2011).

[20] Zaharia, M., Chowdhury, M., Das, T., Dave, A., Ma, J., McCauley, M., Franklin, M.J., Shenker, S. and Stoica, I. 2012. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. *Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation* (2012), 2.

[21] Zaharia, M., Chowdhury, M., Franklin, M.J., Shenker, S. and Stoica, I. 2010. Spark : Cluster Computing with Working Sets. *HotCloud'10 Proceedings of the 2nd USENIX conference on Hot topics in cloud computing* (2010), 10.