APPENDIX A
ARTIFACT DESCRIPTION APPENDIX: MEASURING SWAMPINESS: QUANTIFYING CHAOS IN LARGE HETEROGENEOUS DATA REPOSITORIES

A. Abstract

As scientific data repositories and filesystems grow in size and complexity, they become increasingly disorganized. The coupling of massive quantities of data with poor organization makes it challenging for scientists to locate and utilize relevant data, thus slowing the process of analyzing data of interest. To address these issues, we explore an automated clustering approach for quantifying the organization of data repositories. Our parallel pipeline processes heterogeneous filetypes (e.g., text and tabular data), automatically clusters files based on content and metadata similarities, and computes a novel "cleanliness" score from the resulting clustering. We demonstrate the generation and accuracy of our cleanliness measure using both synthetic and real datasets, and conclude that it is more consistent than other potential cleanliness measures.

B. Description

1) Checklist:

- **Program:** Python 3.6.5
- **Data set:** CDIAC data, synthetic data
- **Run-time environment:** Ubuntu
- **Hardware:** Amazon Web Services virtual machine (128GB ram, 16 cores)
- **Output:** Output files to designated folders outside code directory
- **Experiment workflow:** Clustering pipeline
- **Experiment customization:** Pipeline execution has multiple parameters that a user can manipulate to customize their experiment.
- **Publicly available?:** Code composing the pipeline is available on the following GitHub repository: https://github.com/lollyluann/cluster-datalake

2) How software can be obtained:

- Software dependencies can be obtained using `pip` or some other equivalent installation method
- Code can be forked/cloned from the repository listed.

3) Hardware dependencies:

- 16+ GB of RAM
- To take advantage of parallelized pipeline, multiple cores
- Enough storage for dataset and 20+ GB extra

4) Software dependencies:

- Python 3.x
- Python 3.x packages
  - bs4
  - tqdm
  - numpy
  - pandas
  - matplotlib
  - unipath
  - nltk
  - mpld3
  - sklearn
  - scipy
  - textract
- Linux software
  - gzip
  - swig3.0
  - python3-dev
  - libasound2-dev
  - libpulse-dev
  - calibre
  - gnumeric

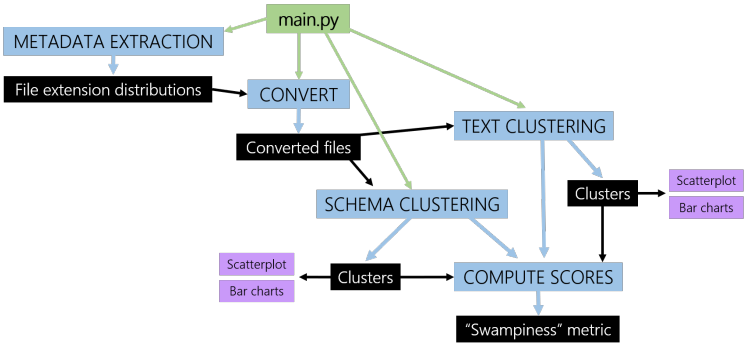
5) Datasets:

- Self-generated toy datasets
  - Code to generate custom versions of these datasets is present on the GitHub repository.
- CDIAC (data from the Carbon Dioxide Information Analysis Center)
  - Transferred using Globus
- pub8 (a subset of the CDIAC dataset)

C. Installation

- Software dependency installation depends on system used.
- For Ubuntu systems, a file is present in the repository containing commands to install all dependencies.
- Installation of the pipeline code can be done through GitHub

D. Experiment workflow



E. Evaluation and expected result

Pipeline generates the following results:

- Metadata information
  - Pie chart regarding filetype distribution
  - Various text files
- Clusters
  - MDS scatterplot
  - Various dependency files
  - Bar charts with file distribution
- Cleanliness score(s)
  - Cleanliness score for each cluster
  - Overall cleanliness for score for entire clustering

  - Best cleanliness score over a range of k

Cleanliness scores should be interpreted as such:

- Score of 1 indicates very clean dataset
- Score of 0 indicates very disorganized dataset
- Score should fall between 0 and 1, inclusive

F. Experiment customization

Our pipeline is constructed in such a way that it possesses multiple parameters that can be manipulated by future users. These allow for customization of the running of the pipeline.

The following parameters are customizable:

- Dataset to run
- Whether to plot extensions
- Whether to convert filetypes
- Whether to cluster tabular data
- Whether to cluster text data
- How many extensions to take
- How many processes to use
- Fill threshold for schema extraction
- Overwrite
  - Tabular distance matrix
  - Tabular plots
  - Text tokens
  - Text clusters
- Whether to use minibatch k-means
- Range of k values to try