A Multi-Level Funneling Approach to Data Provenance Reconstruction

Ailifan Aierken, Delmar B. Davis, Qi Zhang, Kriti Gupta, Alex Wong, Hazeline U. Asuncion
School of Science, Technology, Engineering & Mathematics
University of Washington Bothell
Bothell, WA, USA
{erfan220, davisdb1, qizhang1, kgkriti, aswong89, hazeline}@u.washington.edu

***Abstract*—When data are retrieved from a file storage system or the Internet, is there information about their provenance (i.e., their origin or history)? It is possible that data could have been copied from another source and then transformed. Often, provenance is not readily available for data sets created in the past. Solving such a problem is the motivation behind the 2014 Provenance Reconstruction Challenge. This challenge is aimed at recovering lost provenance for two data sets: one data set (WikiNews articles) in which a list of possible sources has been provided, and another data set (files from GitHub repositories) in which the file sources are not provided. To address this challenge, we present a multi-level funneling approach to provenance reconstruction, a technique that incorporates text processing techniques from different disciplines to approximate the provenance of a given data set. We built three prototypes using this technique and evaluated them using precision and recall metrics. Our preliminary results indicate that our technique is capable of reconstructing some of the lost provenance.**

Keywords- data provenance reconstruction; similarity metrics; semantic analysis; topic modeling; vector space model; longest common subsequence

I. INTRODUCTION

In the course of scientific exploration, simulation, or analysis, various files are created (e.g., input data files, parameter files, intermediate files, output files). Over time, these files accumulate to the point that it becomes difficult for researchers to determine which files are related to each other. In addition, multiple versions of the same files are created due to adjustments in parameters or processing steps. Thus, it becomes difficult for researchers to remember which files contain which changes. On top of this, many of the current provenance techniques assist with recording provenance for current or future analyses, but reconstructing provenance on past analyses without execution logs is difficult.

The 2014 Provenance Reconstruction Challenge [1] was issued to spur researchers to investigate solutions to such a problem. In the challenge, two data sets are provided. We refer to the first set as the human-generated data set, and the other set as the machine-generated data set. Each data set consists of raw data and a provenance ground truth file. For the human-generated data set, the raw data files contain the entire HTML of WikiNews articles [2], with their referenced sources removed, and a list of URIs for related sources. Meanwhile, the ground truth contains the sources removed from the WikiNews articles along with the article filenames. The challenge is to derive the relationship between the source URIs to the correct WikiNews article. For the machine-generated data set, the raw data consists of every version of all files that were ever present in several repositories. Relationships specified in the ground truth, include even those files that no longer exist in the repositories, i.e. deleted files. Moreover, file names are randomized and their timing metadata is removed, in order to simulate provenance loss. The ground truth for the machine-generated data set has been generated from a number of GitHub repositories using the Git2PROV tool [3].

In order to address this challenge, we use a multi-level funneling technique to provenance reconstruction. The funnels group the files based on semantic content and available metadata information. We use multiple levels of funnels in order to accurately group the related files together. Our funnels leverage techniques from research fields such as information retrieval, natural language processing, machine learning, and dynamic programming.

In this paper, we discuss the novelty of our technique and provide some background (Section II), elaborate on our multi-level funneling technique (Section III), present three prototype tools (Section IV), discuss initial results (Section V), and cover current limitations and next steps (Section VI).

II. RELATED WORK AND BACKGROUND

A. Related Work

The idea of reconstructing provenance for a set of files, for which provenance has not been previously recorded, is a largely unexplored area for the data provenance and eScience research communities [1]. Some research groups have started working on techniques for reconstructing provenance. One technique, inspired by the DeepQA approach of IBM Watson [4], analyzes the contents of files and determines the provenance relationships between files using several similarity measures (e.g., text similarity, image similarity, domain-specific similarity, metadata similarity) and edit distance algorithms [5]. Another technique extracts named entities (NEs) and represents these NEs as terms in a Vector Space Model to measure the similarity between files [6]. Both of these techniques assume that timing information (i.e., date of creation) is available for all the files. Our technique, meanwhile, does not require the presence of timing information.

Another reconstruction technique determines file descendant relationships based on research practices and

conventions (e.g., file folder hierarchical structure, locations of experiment file metadata) [7]. This paper, meanwhile, focuses on analyzing the file metadata and the file content (i.e., semantic analysis) to determine provenance relationships between files.

B. Background

Various techniques currently exist to semantically analyze file content. As we discuss later, we use these techniques as funnels in our provenance reconstruction approach. In this section, we provide a high level overview of the various disciplines from which these techniques were gleaned.

Information retrieval (IR) techniques are generally used to retrieve documents that are closely related to a query term or a query phrase [8]. Often times, the retrieved documents are ranked based on a similarity metric.

Machine learning (ML) techniques, meanwhile, generally fall into two categories: supervised and unsupervised techniques [9]. Supervised techniques require training labels whereas unsupervised techniques do not require training labels. Unsupervised techniques can be used to cluster files based on term occurrences.

While IR and ML techniques determine similarity between files based on term occurrences, natural language processing (NLP) takes a step further and analyzes the syntax (phrase structure) and semantics of a document [10]. Thus, NLP techniques can retrieve documents based on meaning rather than simply keywords.

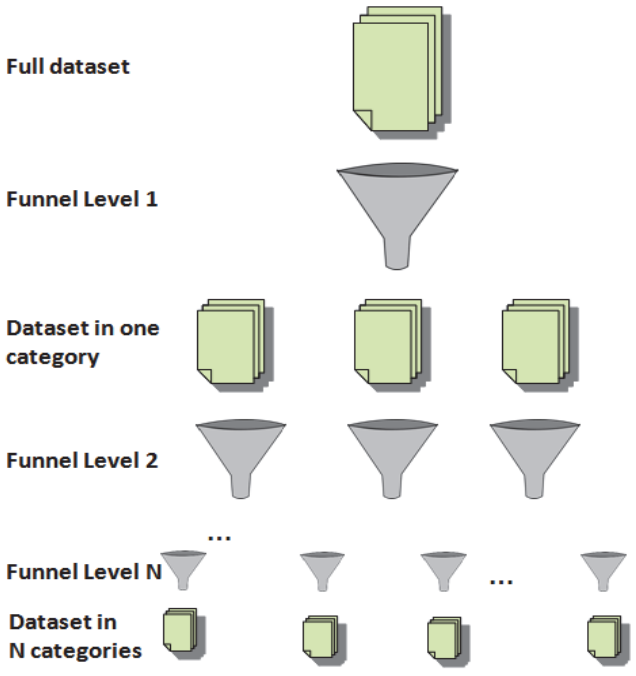

Figure 1. Multi-level Funneling Technique for Provenance Reconstruction

Dynamic programming techniques, such as the longest common subsequence (LCS), can detect the occurrences of string patterns within a given string [11]. Whereas NLP examines the syntax based on language constructs, dynamic programming can detect similarity between files based on the sequence of occurrences of terms, regardless of the language used.

III. Technique Overview

This section covers the main steps of our technique.

A. Pre-process files

The pre-processing step includes obtaining all the relevant files to analyze (e.g., downloading HTML source files for the human-generated data set) and extracting the plain text of each file. The pre-processing step also removes stop words (i.e., words that do not contribute to the semantic content of files such as "a", "an", "the"), punctuation, and tags (for the HTML and Tex files). Pre-processing occurs prior to any application of funnels.

B. Apply one or more funnels to categorize files

We use funnels to categorize files into separate groups. Multiple levels of funnels are applied in order to group files that have provenance relationships into categories (see Figure. 1). The funnel techniques we discuss below are techniques we used thus far.

1) Funnel from Information Retrieval Techniques: Vector Space Model

One particular IR technique, Vector Space Model, represents both query terms and documents as vectors. A term-document matrix is created, with each document represented as a vector of terms. There are different ways to measure the weight of each term, such as term frequency or tf-idf [12]. There are also various similarity metrics (e.g., L1 distance, Euclidian distance, Cosine distance) used to determine the similarity of a query term or query phrase to the given document.

Modeling the corpus into a vector space allows the application of similarity metrics to document-pairs. In the information retrieval techniques funnel, we apply both the L1 norm and cosine similarity metrics. As distances are calculated between document-pairs, files are funneled into distance-based categories.

2) Funnel from Machine Learning Techniques: Topic Modeling

One particular ML technique, topic modeling, is an unsupervised statistical approach for learning semantic topics from a set of documents. The topic modeling technique we use, Latent Dirichlet Allocation (LDA) [13][14], only requires the user to specify the number of topics to learn and the number of iterations to run. With LDA, topics are modeled as a probability distribution over words while documents are modeled as a probability distribution over topics.

We use LDA to funnel files into categories based on similar top *n* topics or similar topic distributions.

3) Funnel from Dynamic Programming Techniques: Longest Common Subsequence

In our approach, we use the LCS algorithm to find the longest common word sequence whose order is the same in both files, which are then used to calculate similarity.

To calculate the similarity, we use the following similarity metric:

Similarity = (number of words in the longest common subsequence of two files) / (the total number of words in the file with less words)

For example, consider the following files:

File1 = "This is a big apple"

File2 = "This apple is very big one"

The number of words in the longest common subsequence is 3 (This is big). The number of words in smaller size of two files is 5 (This is a big apple). The similarity is 3/5 or 60%. The similarity value will be between 0 and 1. We funnel files with a similarity metric close to 1 into one category.

C. Create provenance relationships within categories

Once the files are in their respective groups or categories, we then create a provenance graph based on previously calculated similarity measures. The provenance graph is expressed in the W3C PROV format [15].

D. Evaluate reconstructed provenance

Precision (also called positive predictive value) is the fraction of retrieved instances that are relevant, while recall (also known as sensitivity) is the fraction of relevant instances that are retrieved [12]. Both precision and recall are therefore based on an understanding and measure of relevance.

We use precision and recall to evaluate the performance of our provenance reconstruction tools based on the comparison of the ground truth and the reconstructed provenance.

Precision and recall are then defined as:

Precision = tp / (tp + fp)

Recall = tp / (tp + fn)

where

tp – true positive (file relationship that is found in both the ground truth and the reconstructed provenance)

fp – false positive (file relationship that is found in the reconstructed provenance but not in the ground truth)

fn – false negative (file relationship that is in the ground truth but not found in the reconstructed provenance)

IV. Overview of Prototypes

A. JDAK

JDAK, implemented in Java, has been designed with extensibility in mind. Funnels can be swapped in and out with relative ease, as long as the expected input and output remains consistent. JDAK uses JSoup to assist with pre-processing and Apache Jena to assist with the provenance graph reconstruction.

TABLE I. Provenance Reconstruction Prototypes

Tool	Human-generated	Machine-generated
JDAK	VSM with L1 distance	Metadata (File type), TM
ProvIT	TM, LCS	Metadata (File type), TM, LCS
TraceBack	Stemming, VSM with Cosine Similarity, TM	Metadata (File type), TM

For similarity analysis, JDAK uses a floating distance threshold for the human-generated data set, based on the average distance between a given file and all other possibly related files. For the machine-generated data set, files with a threshold of 85% topic similarity are categorized in the same group. This threshold may be a bit high since JDAK was unable to identify related files with this data set.

B. ProvIT

ProvIT, meanwhile, is unique from the other prototypes in that it is the only tool that uses LCS along with topic modeling as a funnel. For the machine-generated data set, it was necessary to first classify the files into their file types prior to running the topic modeling and the LCS.

C. TraceBack

TraceBack is the only tool that is able to reconstruct provenance for the machine-generated data set. After TraceBack applies the topic modeling funnel, the tool assumes that files are chronologically ordered based on file size (i.e., a file with a smaller file size implies it is created at an earlier time than a file with a bigger file size).

V. Initial Results

To calculate precision and recall, we followed these steps.

Human-generated data set: We trained our models on the entire human-generated data set provided by the challenge. Following the same procedure as the challenge dataset, we created another data set and ground truth from a different set of WikiNews articles with about the same number of articles. We calculated precision and recall on this new data set.

Machine-generated data set: We split the machine-generated ground truth provided by the challenge into two sections: one section consisting of 90% of the ground truth and the other section consisting of the remaining 10%. The 90% section is used to train our models. The 10% section is then used as a hold-out set to calculate precision and recall.

TABLE II. Precision and Recall for each Prototype

Tool	Human-generated	Machine-generated
JDAK	Precision: 23% Recall: 45%	Not available
ProvIT	Precision: 77% Recall: 47%	Precision: < 1% Recall: < 1%
TraceBack	Not available	Precision: 78% Recall: 68%

Table II shows initial precision and recall results for our three prototypes. Note that the precision and recall numbers for the machine-generated data set for TraceBack apply to a major subset of the files (.tex and .pdf files, which comprise 81% of all the files).

VI. Limitations and Next Steps

In this section, we discuss the current limitations of our technique.

A. Order of file creation

It is a challenge to determine the order of file creation solely from the semantic content of files. For the human-generated data set, the list of possible sources have been provided, which implies the order of file creation. For the machine-generated files, however, such source information was not provided. Solely analyzing the content of the file does not indicate which files were created first (because the differences between the files may either be an addition or a deletion). Thus, it is necessary to have additional metadata information to determine file derivation relationships.

B. Other file types

Thus far, we focused our analysis on text-based files (e.g., HTML files, Tex files). In the future, we plan to also analyze image files using topic modeling [13].

C. Time constraints for tool development

Due to time constraints and implementation issues encountered, we were not able to obtain acceptable precision and recall numbers for some of the datasets. We anticipate that we will achieve better results as we continue to refine our tools in the future.

D. Generalizability of the technique for other data sets

Since we have not tried our prototypes on other types of human-generated or machine-generated data sets, we cannot currently assess the generalizability of our technique. We plan to use other data sets, including data sets with other file extensions. Thus far, ProvIT is able to handle 26 file types, including .bib, .pdf, .py, .tex, and .xml files.

VII. Conclusion

In this paper, we present a systematic technique for reconstructing or recovering provenance for experiment or analysis files that have been generated in the past. Our multi-level funneling technique combines the strengths of text processing techniques from various areas such as information retrieval, machine learning, natural language processing, and dynamic programming. We evaluated our three prototypes using precision and recall metrics. Our initial results indicate that our technique is able to partially reconstruct provenance for the human-generated and machine-generated data sets.

Acknowledgment

We would like to thank Priyadharshini Agoramoorthy, Nita Y. Karande, Amrita Kaur, Bang Liu, Anshu Priyadarshini, Prachi Singh, and Jason W. Woodring for their assistance in tool development. This work is based in part upon work supported by the US National Science Foundation under Grant No. ACI 1350724. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the NSF.

References

[1] Data2Semantics, "Provenance reconstruction challenge 2014." http://www.data2semantics.org/prov-reconstruction-challenge/, June 2014.

[2] Wikimedia Foundation Inc., "Wikinews." http://en.wikinews.org/, June 2014.

[3] iMinds and Data2Semantics, "Git2prov." http://git2prov.org/, June 2014.

[4] D. Ferrucci, E. Brown, J. Chu-Carroll, J. Fan, D. Gondek, A. A. Kalyanpur, A. Lally, J. W. Murdock, E. Nyberg, J. Prager, N. Schlaefer, and C. Welty, "Building watson: An overview of the DeepQA project," AI magazine, vol. 31, no. 3, pp. 59–79, 2010.

[5] S. Magliacane, "Reconstructing provenance," in Proc of International Semantic Web Conference, vol. 7650 of Lecture Notes in Computer Science, pp. 399–406, Springer Berlin Heidelberg, 2012.

[6] T. De Nies, S. Coppens, D. Van Deursen, E. Mannens, and R. Van de Walle, "Automatic discovery of high-level provenance using semantic similarity," in Provenance and Annotation of Data and Processes, vol. 7525 of Lecture Notes in Computer Science, pp. 97–110, Springer Berlin Heidelberg, 2012.

[7] D. B. Davis, H. U. Asuncion, G. M. Abdulla, and C. W. Carr, "Towards recovering provenance with experiment explorer," Proc of the IARIA Conference on Information, Process, and Knowledge Management, pp. 104–110, 2013.

[8] V. V. Raghavan and S. M. Wong, "A critical analysis of vector space model for information retrieval," Journal of the American Society for Information Science, vol. 37, no. 5, pp. 279–287, 1986.

[9] D. J. Hand, H. Mannila, and P. Smyth, Principles of Data Mining. MIT Press, 2001.

[10] D. Jurafsky and J. H. Martin, Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. Prentice Hall, 2008.

[11] R. Bellman, Dynamic Programming. Princeton University Press, 1957.

[12] C. D. Manning, P. Raghavan, and H. Schütze, Introduction to Information Retrieval, vol. 1. Cambridge University Press, 2008.

[13] D. M. Blei, "Probabilistic topic models," Communications of the ACM, vol. 55, pp. 77–84, Apr. 2012.

[14] A. Asuncion, M. Welling, P. Smyth, and Y. W. Teh, "On smoothing and inference for topic models," Uncertainty in Artificial Intelligence., 2009.

[15] W3C. http://www.w3.org/TR/2013/NOTE-prov-overview-20130430/.