An Analytical Review on the Techniques Opted For The Detection of Cloning in Spread Sheets

Mitali

Abstract: Spreadsheets are widely used in industry: it is estimated that end-user programmers outnumber programmers by a facto . However, spreadsheets are error-prone, numerous companies have lost money because of spreadsheet errors. One of the causes for spreadsheet problems is the prevalence of copy-pasting. This paper focuses on the methods of identifying data clone in spread sheets and their efficiency. The paper also presents suitable algorithms for data cloning in the data mining region.

Index Terms : Data Clone , Data Mining , Fatal Errors, Spread Sheets.

I. INTRODUCTION

Spreadsheets are widely used in industry: it is estimated that end-user programmers outnumber programmers by a factor [1]. However, spreadsheets are error-prone, numerous companies have lost money because of spreadsheet errors. One of the causes for spreadsheet problems is the prevalence of copy-pasting. Based on existing text-based clone detection algorithms, different developed algorithms have been designed to detect data clones in spreadsheets: formulas whose values are copied as plain text in a different location. The results of the evaluation clearly indicate that

1) Data clones are common,

2) Data clones pose threats to spreadsheet quality

The data are cloned by including multiple copies of the encounter histories, i.e., duplicating the encounter histories. In MARK, all that needs to be done is to multiply the encounter history frequencies of each group by the number of clones desired [1]. Consider the example of cloning the data 100 times. An encounter history for an analysis with 2 groups and no individual covariates that looks like this:1100101001032; could be cloned 100 times by entering the following encounter history: 11001010010300200;By cloning the data, the sample size is increased without changing the parameter estimates [2]. So, if the original estimates are compared to the cloned estimates, the values of the estimates will remain the same for parameters that are not confounded and are otherwise properly estimated. However, because the sample size has been increased, the standard errors of the cloned estimates will be smaller than the original standard errors. The expected result for parameters that are estimable is SE(original) = SE(cloned)*sqrt(number of clones). [2] As an example, if the data are cloned 100 times, then the standard errors of the cloned data will be 1/10 of the original standard errors.

Manuscript published on 30 November 2013.
***Correspondence Author(s)**
Mitali, M-Tech– Cse Rimt, Mandi Gobindgarh, Punjab, India.

II. TYPES OF DATA CLONING

A) Regular: In a regular clone entire data from one spread sheet can be copied to another data format or to another spread sheet itself. This cloning method is done if the data is getting updated in the industry and the concerned person wants to keep the previous and the new record all together.

B) Irregular: Irregular data clone is very often in the industry as to avoid work and to save time . Industry people copy the database from one organization to another to build up links with the clients already associated with the first organization. [3]

III. METHODS AND ALGORITHM FOR CLONE DETECTION

There are several algorithms in this contrast to identify the data clone and the percentage in which they have been copied. Here is a review of some of the finest algorithms.

A. ICA (Increment Component Analysis)

They employ a generalized suffix-tree that can be updated efficiently when the source changes [4]. The amount of effort required for the update only depends on the size of the change, not the size of the code base. Unfortunately, generalized suffix-trees require substantially more memory than read-only suffix-trees, since they require additional links that are traversed during the update operations. Since generalized suffix-trees are not easily distributed across different machines, the memory requirements represent the bottleneck with respect to. scalability. Consequently, the improvement in incremental detection comes at the cost of substantially reduced scalability.

B. AST Based Incremental Method

Nguyen et al. presented [5] an AST-based incremental approach that computes characteristic vectors for all sub trees of the AST for a file. Clones are detected by searching for similar vectors. If the analyzed data changes, vectors for modified files are simply recomputed. As the algorithm is not distributed, its scalability is limited by the amount of memory available on a single machine. A related approach that also employs AST sub tree hashing is proposed by Chilowicz et al.[5]. However, such systems often contain substantial amounts of cloning [3] making clone management for them especially relevant. Instead, his approach does not require a parser.

C. Neural Logistics for Clone Detection

The neural networks are non-linear statistical data modeling tools that are inspired by the functionality of the human brain using a set of interconnected nodes [6]. Neural networks are widely applied in classification and clustering, and its advantages are as follows.

First, it is adaptive; second, it can generate robust models; and third, the classification process can be modified if new training weights are set. Neural networks are chiefly applied to credit card spread sheet data, automobile insurance spread sheet data and corporate fraud . Literature describes that neural networks can be used as a financial fraud detection tool. The neural network fraud classification model employing endogenous financial data created from the learned behavior pattern can be applied to a test sample. The neural networks can be used to predict the occurrence of corporate fraud at the management level [7].

D .Text Based Technique

Text-based techniques perform little or no transformation to the raw source data of spread sheet before attempting to detect identical or similar (sequences of) data. [8]

E. Token Based Technique

Token-based techniques apply a lexical analysis (tokenization) to the source code and, subsequently, use the tokens as a basis for clone detection .[9]

F. PDG-Based Approach

PDG-based approaches go one step further in obtaining a source code representation of high abstraction. Program dependence graphs (PDGs) contain information of a semantical nature, such as control and data flow which look for similar sub graphs in PDGs in order to detect similar data. It first augments a PDG with additional details on expressions and dependencies, and similarly applies an algorithm to look for similar subgraphs [10]

REFERENCES

1. H. A. Basit, D. C. Rajapakse, and S. Jarzabek. Beyond templates: a study of clones in the STL and some general implications. In Proc. of the Int'l Conf. on Software Engineering, pages 451{459, 2005.
2. I. D. Baxter, A. Yahin, L. M. de Moura, M. Sant'Anna, and L. Bier. Clone detection using abstract syntax trees. In Proc. of the Int'l Conf. on Software Maintenance, pages 368{377, 1998.
3. K. Beck. extreme Programming explained, embrace change. Addison-Wesley, 2000.
4. " Hang Dai and Jingshi He Dongguan 523808", China Research Journal of Applied Sciences, Engineering and Technology 6(5): 895-899, 2013 ISSN: 2040-7459; e-ISSN: 2040-7467 2013
5. T. T. Nguyen, H. A. Nguyen, J. M. Al-Kofahi, N. H. Pham, and T. N. Nguyen, "Scalable and incremental clone detection for evolving software," ICSM'09, 2009.
6. Ghosh, S., & Reilly, D. L. (1994). Credit card fraud detection with a neural-network, 27th Annual Hawaii International, Conference on System Science 3 (1994) 621–630.
7. Beasley, M. (1996). An empirical analysis of the relation between board of director composition and financial statement fraud. The Accounting Review, 71(4), 443–466.
8. J. H. Johnson, "Identifying redundancy in source code using fingerprints,"in Proc. of CASCON '93, 1993, pp. 171–183.
9. M. Fisher and G. Rothermel, "The EUSES spreadsheet corpus: a shared resource for supporting experimentation with spreadsheet dependability mechanisms," ACM SIGSOFT Software Engineering Notes, vol. 30, no. 4, pp. 1–5, 2005.
10. I. D. Baxter, A. Yahin, L. M. de Moura, M. Sant'Anna, and L. Bier, "Clone detection using abstract syntax trees," in Proc. of ICSM '98, 1998, pp. 368–377.
11. R. Komondoor and S. Horwitz, "Using slicing to identify duplication in source code," in Proc. of SAS '01, 2001, pp. 40–56.