Towards Automated Detection of Qualitative Spreadsheet Errors in Multi-user Environments

Miro Zdilar
University of Zagreb
Faculty of Organization and Informatics
Pavlinska 2, 42000 Varaždin, Croatia
miro.zdilar@gmail.com

Abstract. *Spreadsheets are one of the most used software systems in business and academia. Since the first introduction of electronic spreadsheets for personal computers in 1979, they have evolved into a powerful computing platform. Their use spans from asset administration to complex scientific analysis and process modelling. However, spreadsheets are associated with high incidence of errors, causing companies and organizations significant reputational and financial losses. The goal of this work-in-progress article is to present a model for an automated detection of qualitative spreadsheet errors in multi-user environments, based on abstract state machines.*

Keywords. Spreadsheets, Spreadsheet Errors, Automated Error Detection, Qualitative Spreadsheet Errors in Multi-User Environments, Spreadsheet Quality Assurance

1 Introduction

Spreadsheets are widely used and can be considered as one of the most successful end-user programming systems. End-user programming systems allow end-users to build and execute powerful computer programs without the use of traditional programming languages and supporting development tools. It has been estimated that the number of end-user programmers outnumber traditional software programmers (Scaffidi et al., 2005). Spreadsheets are used in almost all companies in the US and Europe for financial reporting and strategic planning (Panko & Ordway, 2008). The framework for the identification of spreadsheet usage patterns (Reschenhofer & Matthes, 2015) identified use cases for the following business processes; capacity planning, financial reporting, stakeholder analysis, risk management, performance calculation, data transformation, cash-flows analysis, time-series transformations, and simulations. Despite their great success and importance, spreadsheets are known to be error prone (Rajalingham et al., 2000). The European Spreadsheet Risk Interest Group (EuSpRIG), a non-profit and voluntary organization, maintains a list of horror stories that illustrate problems with uncontrolled usage of spreadsheets (European Spreadsheet Risk Interest Group, 2023).

This work-in-progress article presents a model for automated detection of qualitative spreadsheet errors in multi-user environments. According to taxonomy of spreadsheet errors (Rajalingham et al., 2008), qualitative errors do not immediately produce incorrect values but degrade the quality of the spreadsheet. Spreadsheet also becomes more prone to misinterpretation and more difficult to maintain. The rest of this paper is organized as follows. Section 2 is a short overview of related work with the problem statement in section 3. Afterwards, in section 4, the architecture of the proposed model for automated detection of qualitative spreadsheet errors in multi-user environments is presented. In section 5 we provide initial results of presented model simulations. Finally, in section 6 we discuss results and provide directions for further model evaluation and future research.

2 Related Work

2.1 Taxonomy of Spreadsheet Errors

Omnipresence of spreadsheets and impact of spreadsheet errors triggered a significant interest of the research community. Focus of researchers have been in the area of detection of spreadsheet errors, as well as identification of frameworks and methodologies that should prevent occurrences of spreadsheet errors (Powell et al., 2008). An important aspect of spreadsheet research deals with the development of taxonomies for spreadsheet errors.

Early studies listed types of errors detected without a classification of spreadsheet errors (Brown & Gould, 1987). Galetta et al. (1993) introduced two classes of spreadsheet errors. The authors distinguished between domain errors and device errors. Domain refers to the spreadsheet's application area (e.g., accounting), while

device refers to the spreadsheet technology itself. Authors conducted an experiment with 30 accounting experts and 30 students to seek up to two errors introduced in each of six provided spreadsheets used during the experiment. While accounting experts performed better in detection of domain errors, students demonstrated comparable performance in detection of device errors. For example, a mistake in logic due to a misunderstanding of depreciation is a domain error but entering the wrong reference in the depreciation function SLN is a device error.

In one of the first attempts to offer complete classification of errors, researchers distinguished between quantitative and qualitative errors (Panko & Halverson, 1996). Quantitative errors are related to the current version of a spreadsheet, while qualitative errors refer to risky practices that might lead to an error in later stages of a spreadsheet's lifecycle and degrade the quality of the spreadsheet. Authors divided quantitative errors into three categories:

- Mechanical errors, due to mistakes in typing or pointing
- Logic errors, due to choosing the wrong function or creating the wrong formula
- Omission errors, due to misinterpreting the situation to be modelled

A more comprehensive approach to taxonomy of spreadsheet errors is focused on user-generated errors (Rajalingham et al., 2008). This taxonomy explicitly distinguished between developer and end-user generated spreadsheet errors. End-user errors are further classified into Data Inputter and Interpreter errors.

In recent years, researchers identified the need to relate types and occurrences of spreadsheet errors with quality of spreadsheets. Intuitively, a higher incidence of spreadsheet errors suggests that the overall quality of a spreadsheet is low. O'Beirne (2008) presented an overview of Information Quality and Data Quality within the context of spreadsheets. The author presented a comprehensive list of information quality attributes in the context of spreadsheet programs. In addition, the author presented checks and control procedures for spreadsheet information and quality processes. Cunha at al. (2012) proposed a quality model for spreadsheets with a set of domain specific metrics for spreadsheets, which are used to measure concrete spreadsheet characteristics. The presented quality model for spreadsheets is based on the widely accepted ISO/IEC 9126 international standard for software product quality (International Organization for Standardization, 2001). Authors provided a comprehensive analysis of ISO/IEC 9126 standard and map relevant quality attributes to spreadsheets.

2.2 Spreadsheet Errors Detection

High incidents of spreadsheet errors have led to a series of research and commercial auditing tools. Nixon & O'Hara (2010) provided a structured assessment of several commercial auditing tools. The authors conducted an experiment based on real spreadsheets used for generating turnover reports in a retail company. In total, 17 qualitative and quantitative errors have been added to the reporting spreadsheet used for testing 4 commercial spreadsheet auditing tools and Microsoft Excel built-in auditing functions. Results of the test proved that spreadsheet auditing tools are generally useful as supporting tools for spreadsheet experts. One commercial auditing tools achieved detection rates of over 80%. However, results were largely subjective and limited to only one type of a relatively simple spreadsheet used for financial reporting. Interesting to note is that Microsoft Excel built-in auditing functions only scored 24%.

Another study conducted by Aurigemma & Panko (2010) compared error detection success rate of human expert auditors with two commercial spreadsheet static analysis tools. The corpus of 75 spreadsheets used in the study was created by undergraduate students in an introductory management information system course (Panko, 2010). The overall results indicated that human spreadsheet auditors significantly outperformed automated static analysis tools in the detection of spreadsheets errors generated by humans.

Several commercial and governmental institutions published their own spreadsheet auditing tools and methodologies. Officers of HM Customs and Excise have been performing field audits of taxpayers' spreadsheet applications since 1985. Effectiveness of HM Customs and Excise spreadsheet testing methodology has been evaluated by Butler (2000). This procedure is hybrid and includes manual activities performed by an expert auditor, as well as automated activities performed with commercial software auditing tool. The procedure is based on the identification of high-risk cells and formulas with potentially high financial impact and tax losses. The main contribution of commercial auditing tools is with identification of high risk and complex formulas with many dependencies to linked cells and other data sources. Final assessment on identified high risk cells and formulas are performed by expert human auditors using different testing strategies such as control checks, reviews and reperformance. This methodology is effective for smaller to medium sized spreadsheet models used for tax calculations and reporting.

An automated method to infer data types from a spreadsheet was presented by Erwig & Burnett (2002). The proposed method for inferring types from spreadsheets is based on the concrete notion of units instead of the abstract concept of types. Authors used header information given by spreadsheets to derive units. In continuation to the presented concept around units, Ahmad et al. (2003) developed a type system for statically detecting spreadsheet errors. Author named

the proposed type system for spreadsheets "unit checking". The same researchers also presented a collection of rules for the identification of spreadsheet weaknesses that are likely to be errors. The presented type system for spreadsheets also relies on the concept of the header that defines common unit for groups of cells.

Jannach et al. (2014) conducted comprehensive literature review and proposed two main categories of automated spreadsheet quality assurance approaches:

- "Finding and fixing errors" is about techniques and tools that are mainly designed to help the user detect errors and understand the reasons for the errors.
- "Avoiding Errors" is about techniques and tools that should help developer create error free spreadsheets.

In addition to research related to manual and automated spreadsheet quality assurance, important to note is research related to the automation of spreadsheet testing (Abraham & Erwig, 2008). Modern approaches to spreadsheet development follow good practices and methodologies for software development life cycle and the automation of spreadsheet testing is valuable both for researchers and industry practitioners. To minimise bias and ensure validity of test cases, the authors developed mutation operators for spreadsheets. Authors followed original concepts of mutation testing developed for general purpose programming languages.

Recent research and studies are focused on the application of large language models to spreadsheet environments. A team of researchers from Microsoft Corporation presented the FLAME language model for spreadsheet formulas (Joshi et al., 2023). FLAME uses the Microsoft Excel specific formula tokenizer and other techniques to achieve competitive performance with a substantially smaller model (60 million parameters) and two order of magnitude less training data, compared to other large language models such as Codex. Researchers used a training dataset of 972 million formulas extracted from a corpus of 1,8 million Excel workbooks. FLAME was evaluated on three different tasks for Excel formulas: last-mile repair, autocompletion and syntax reconstruction. Syntax reconstruction is a novel term coined by the authors of FLAME, and the goal of this task is to reconstruct original Excel formula from formulas with removed delimiters. The presented FLAME language model outperforms larger language models, such as Codex-Davinci (175 billion parameters), Codex-Cushman (12 billion parameters), and CodeT5 (220 million parameters), in 6 out of 10 experimental settings.

3 Problem Statement

Our proposed model for automated detection of qualitative spreadsheet errors in multi-user environments will address the following spreadsheet errors:

- Noncompliance and deviations from spreadsheet development guidelines (Esch et al. 2010).
- Unauthorized changes and modifications to spreadsheet programs and data during all stages of a spreadsheet's lifecycle.

Proposed model for automated detection of qualitative spreadsheet errors in multi-user environments, falls into both categories of automated spreadsheet quality assurance approaches developed by Jannach et al. (2014). It allows users to find qualitative spreadsheet errors as well assists spreadsheet developers in creation of error free spreadsheets.

Qualitative spreadsheet errors are difficult to identify and troubleshoot in multi-user environments due to frequent organizational changes in users' authorizations and complex development standards with codified rules for spreadsheet development. In the following we list a few practical examples of qualitative spreadsheet errors according to guidelines for the development and validation of spreadsheets (Esch et al. 2010):

- String literals has been entered for nominal volume in input worksheet. Guidelines for the development require that only numerical values are allowed for nominal volume.
- Inventory number of the asset is not visible on screen and on the print-out. Guidelines for the development require that inventory asset number is visible on screen and on print-out.
- Additional sample numbers have been added to input worksheet and validation range was overwritten. Guidelines for the development require that only limited number of sample numbers are allowed in input worksheet according to predefined range of input values.
- Calculation for sample standard deviation was changed by unauthorized spreadsheet end user. Guidelines for the development require that changes to calculations are performed only by authorized users.

4 Model Architecture

Bellow we provide a conceptual model for automated detection of qualitative spreadsheet errors in multi-user environments based on the application of Abstract State Machines (ASM) (Gurevich, 2000). The following notation is used to describe data flow and interaction between model components and spreadsheet users:

- S_i: State of spreadsheet at time i
- A_{Si}: State of ASM at time i
- B_{U_n}: Behaviours B of user U_n

- $\Delta S_i(B_{U_n})$: Spreadsheet state transitions between state S_i and state S_{i-1}, caused by behaviours B of user U_n
- S_e: Final state of spreadsheet (end of spreadsheet lifecycle)

The core idea for the proposed model for automated detection of spreadsheet errors is based on equivalence between different lifecycle states of a spreadsheet program and the corresponding automatically synthesized ASM. As depicted visually in Figure 1, the state of spreadsheet S corresponds to an equivalent state of ASM A_S. During the lifecycle, a spreadsheet might transition through m transitions to reach state S_i. The corresponding ASM should follow state transitions of spreadsheet S_i to reach state A_{Si}.

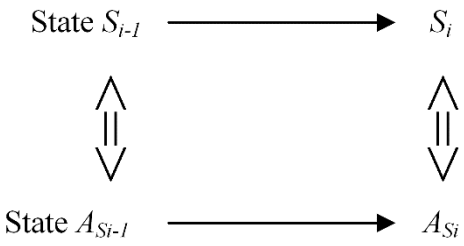

Figure 1. Spreadsheet and ASM state equivalence

Using the same notation, spreadsheet stages during different phases of the lifecycle are modelled by a finite sequence of state transitions as follows:

$$S_0 \xrightarrow{\Delta S_0(B_{U_n})} S_1 \xrightarrow{\Delta S_1(B_{U_n})} S_2 \xrightarrow{\Delta S_2(B_{U_n})} \dots S_e \quad (1)$$

Where S_0 is the initial state of the spreadsheet ("first creation") and S_e is the final state of spreadsheet ("end of lifecycle"). $\Delta S_i(B_{U_n})$ are transitions between spreadsheet states caused by behaviours B of user U_n. When required by the spreadsheet user, at each state of S_i where $i \in \{0,1,2,..,e\}$, the equivalent state A_{Si} for ASM is synthesized in order to determine $\Delta S_{i-1}(B_{U_n})$.

Figure 2 illustrates a high-level component diagram and interaction of users for presented conceptual model.

Data processing within proposed model for automated detection of qualitative spreadsheet errors is performed in three consecutive steps. During first step, spreadsheet submitted by user as input to proposed model is parsed with dedicated parser structured around spreadsheet formula language grammar. We implemented a spreadsheet parser in Python 3 programming language (Van Rossum & Drake, 2009). We utilized the OpenPyXL library for manipulating spreadsheet files and the Tokenizer module for tokenizing spreadsheet formulae (Zumstein, 2021). As a result of successful spreadsheet parsing, a directed graph is generated with hierarchical representation of all resources that constitute the input spreadsheet. During second step, based on graph representation of spreadsheet, abstract state machine is generated. Structure and properties of graph components in synthesized abstract state machine are determined with constitutive elements of spreadsheet programs parsed during the first step of data processing. During the third step of data processing within proposed model, synthesized abstract state machine are explored to detect structures that correspond to spreadsheet errors. We utilized the NetworkX python library (Hagberg et al. 2008) to generate and query a directed graph with the representation of the abstract state machine and associated transition states.

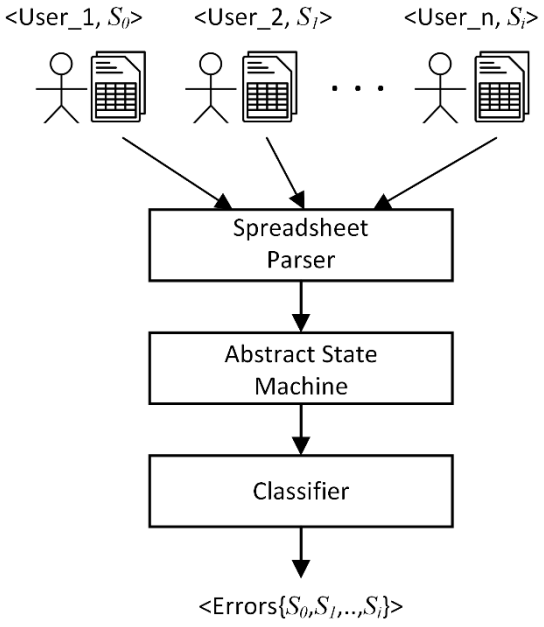

Figure 2. High-level conceptual model

5 Model Evaluation

As this research is work-in-progress, we explore following initial research question:

- RQ: How does the proposed model perform in detection of unauthorized spreadsheet changes performed by spreadsheet users?

To evaluate research question, we randomly assigned different user roles to 15 experienced spreadsheet users. Details of designed user roles and permissions are listed in Table 1.

We created an initial error free spreadsheet template with a sales report for 10 markets and 5 products. Input data was presented in the INPUT worksheet. A model to calculate total sales per product and per market was developed in the MODEL worksheet. Results were presented in the OUTPUT worksheet with references to the calculated model. The TESTING worksheet had a simple testing procedure and empty named ranges to assist the Tester with documenting the performed testing. This initial spreadsheet template was used to generate an

equivalent abstract state machine representation with proposed model for automated detection of qualitative spreadsheet errors.

Table 1. Assigned spreadsheet user roles and permissions

User role	Permissions
Developer	1. Create and modify all resources in MODEL worksheet. 2. Utilizes all data from INPUT worksheets. 3. Create and format results in OUTPUT worksheet.
Tester	1. Create and modify all resources in TESTING worksheet.
Data Inputter	1. Modify cell values in INPUT worksheet.
Owner	1. Modify cell values in INPUT worksheet. 2. Modify named ranges in all worksheets.

Without informing the 15 expert spreadsheet users about their roles, we requested them to document and perform 7 different changes in the given spreadsheet template that corresponds to permissions listed in Table 1. Based on spreadsheets provided by expert participants, we generated corresponding abstract state machines with our proposed model. In order to identify changes performed by users we queried all edges of the synthesized abstract state machines. Edges in graph representation of abstract state machines represents state transitions and, in our model, corresponds to changes and modifications to spreadsheets performed by users. We compared results of graph queries and detected graph changes with assigned users' roles and list of actual changes performed by spreadsheet users.

Our model correctly identified all changes and modifications performed by spreadsheet users. Dictionaries generated as the result of query with NetworkX python package were complex and difficult to analyse, and we utilized a simple script to transform them into tabular format. In addition, we manually reviewed all query result in tabular format with participating spreadsheet experts. Even though format of generated queries is difficult to interpret by human reviewers, our model and corresponding abstract state machines correctly detected all changes introduced by spreadsheet users.

6 Conclusion and Future Research

In this paper we have presented initial evaluations of our model for automated detection of qualitative spreadsheet errors. We developed the presented model with the goal to address specific classes of spreadsheet qualitative errors that are difficult to detect in multi-user environments. As this research is work-in-progress, our initial evaluations are limited to a smaller population of spreadsheets and users. However, presented results are helpful and provide valuable insights to model capabilities.

In our future research, we will focus on performance improvement of spreadsheet parsing and data structures for abstract state machines representation. We will further evaluate proposed model in more comprehensive case studies and large multi-user environments. We will also explore opportunities to implement machine learning algorithms for the detection of spreadsheet qualitative errors in multi-user environments.

Acknowledgments

The author would like to thank Prof. dr. sc. Markus Schatten from Artificial Intelligence Laboratory, for valuable comments and suggestions.

References

Abraham, R., & Erwig, M. (2008). Mutation operators for spreadsheets. *IEEE Transactions on Software Engineering*, *35*(1), 94-108.

Ahmad, Y., Antoniu, T., Goldwater, S., & Krishnamurthi, S. (2003). A type system for statically detecting spreadsheet errors. *In 18th IEEE International Conference on Automated Software Engineering, 2003. Proceedings.* (pp. 174-183). IEEE.

Aurigemma, S., & Panko, R. R. (2010). The detection of human spreadsheet errors by humans versus inspection (auditing) software. *arXiv preprint arXiv:1009.2785*.

Brown, P.S. and Gould, J.D., (1987). An experimental study of people creating spreadsheets. *ACM Transactions on Information Systems (TOIS)*, *5*(3), pp.258-272.

Butler, R. J. (2000). Is this spreadsheet a tax evader? How HM Customs and Excise test spreadsheet applications. In *Proceedings of the 33rd Annual Hawaii International Conference on System Sciences* (pp. 6-pp). IEEE.

Cunha, J., Fernandes, J. P., Peixoto, C., & Saraiva, J. (2012). A quality model for spreadsheets. In *2012 Eighth International Conference on the Quality of Information and Communications Technology* (pp. 231-236). IEEE.

Erwig, M., & Burnett, M. (2002). Adding apples and oranges. In *PADL* (Vol. 2, pp. 173-191).

Esch, P. M., Moor, C., Schmid, B., Albertini, S., Hassler, S., Donzé, G., & Saxer, H. P. (2010).

Good Laboratory Practice (GLP)–Guidelines for the Development and Validation of Spreadsheets. *The Quality Assurance Journal, 13*(3-4), 41-56.

European Spreadsheet Risk Interest Group. (2023). EuSpRIG Horror Stories. https://eusprig.org/research-info/horror-stories/ (May 30, 2023)

Galletta, D. F., Abraham, D., El Louadi, M., Lekse, W., Pollalis, Y. A., & Sampler, J. L. (1993). An empirical study of spreadsheet error-finding performance. *Accounting, Management and Information Technologies*, *3*(2), 79-95.

Gurevich, Y. (2000). Sequential abstract state machines capture sequential algorithms, In *ACM Transactions on Computational Logic* (Vol. 1, No. 1, pp.77–111). ACM.

Hagberg, A., Swart, P., & S Chult, D. (2008). *Exploring network structure, dynamics, and function using NetworkX* (No. LA-UR-08-05495; LA-UR-08-5495). Los Alamos National Lab.(LANL), Los Alamos, NM (United States).

International Organization for Standardization. (2001). *Software engineering-product quality-part 1: Quality model* (ISO Standard No. ISO/IEC 9126-1).

Jannach, D., Schmitz, T., Hofer, B., & Wotawa, F. (2014). Avoiding, finding and fixing spreadsheet errors–a survey of automated approaches for spreadsheet qa. Journal of Systems and Software, 94, 129-150.

Joshi, H., Ebenezer, A., Cambronero, J., Gulwani, S., Kanade, A., Le, V., ... & Verbruggen, G. (2023). FLAME: A small language model for spreadsheet formulas. *arXiv preprint arXiv:2301.13779*.

Nixon, D., & O'Hara, M. (2010). Spreadsheet auditing software. *arXiv preprint arXiv:1001.4293*.

O'Beirne, P. (2008). Information and data quality in spreadsheets. *arXiv preprint arXiv:0809.3609*.

Panko, R. R. (2000). Two corpuses of spreadsheet errors. In *Proceedings of the 33rd Annual Hawaii International Conference on System Sciences* (pp. 8-pp). IEEE.

Panko, R. R., & Halverson, R. P. (1996). Spreadsheets on trial: A survey of research on spreadsheet risks. In *Proceedings of HICSS-29: 29th Hawaii International Conference on System Sciences* (Vol. 2, pp. 326-335). IEEE.

Panko, R. R., & Ordway, N. (2008). Sarbanes-oxley: What about all the spreadsheets*?*. *arXiv preprint arXiv:0804.0797*.

Powell, S. G., Baker, K. R., & Lawson, B. (2008). A critical review of the literature on spreadsheet errors. *Decision Support Systems, 46*(1), 128-138.

Rajalingham, K., Chadwick, D. R., & Knight, B. (2008). Classification of spreadsheet errors. *arXiv preprint arXiv:0805.4224*.

Rajalingham, K., Chadwick, D., Knight, B., & Edwards, D. (2000). Quality control in spreadsheets: a software engineering-based approach to spreadsheet development. In *Proceedings of the 33rd Annual Hawaii International Conference on System Sciences (pp. 10-pp). IEEE.*

Reschenhofer, T., & Matthes, F. (2015). A Framework for the Identification of Spreadsheet Usage Patterns. In *ECIS.*

Scaffidi C., Shaw M., Myers B. A. (2005). Estimating the numbers of end users and end users programmers, In *Proc. of VL/HCC '05, pp. 207-214.*

Van Rossum, G., & Drake, F. L. (2009). *Python 3 Reference Manual.* Scotts Valley, CA: CreateSpace.

Zumstein, F. (2021). *Python for Excel: A Modern Environment for Automation and Data Analysis* (pp. 155-179). O'Reilly Media.