VEnron: A Versioned Spreadsheet Corpus and Related Evolution Analysis

Wensheng Dou[1], Liang Xu[1], Shing-Chi Cheung[2], Chushu Gao[1], Jun Wei[1], Tao Huang[1]
[1]State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing, China
[2]Dept. of Computer Science and Engineering, The Hong Kong University of Science and Technology, Hong Kong, China
[1]{wsdou, xuliang12, gaochushu, wj, tao}@otcaix.iscas.ac.cn, [2]scc@cse.ust.hk

ABSTRACT

Like most conventional software, spreadsheets are subject to software evolution. However, spreadsheet evolution is rarely assisted by version management tools. As a result, the version information across evolved spreadsheets is often missing or highly fragmented. This makes it difficult for users to notice the evolution issues arising from their spreadsheets.

In this paper, we propose a semi-automated approach that leverages spreadsheets' contexts (e.g., attached emails) and contents to identify evolved spreadsheets and recover the embedded version information. We apply it to the released email archive of the Enron Corporation and build an industrial-scale, versioned spreadsheet corpus *VEnron*. Our approach first clusters spreadsheets that likely evolved from one to another into *evolution groups* based on various fragmented information, such as spreadsheet filenames, spreadsheet contents, and spreadsheet-attached emails. Then, it recovers the version information of the spreadsheets in each evolution group. VEnron enables us to identify interesting issues that can arise from spreadsheet evolution. For example, the versioned spreadsheets popularly exist in the Enron email archive; changes in formulas are common; and some groups (16.9%) can introduce new errors during evolution.

According to our knowledge, VEnron is the first spreadsheet corpus with version information. It provides a valuable resource to understand issues arising from spreadsheet evolution.

CCS Concepts

•Applied computing→Computers in other domains→Personal computers and PC applications→Spreadsheets •Software and its engineering → Software creation and management → Software post-development issues→Software reverse engineering.

Keywords

Version; spreadsheet; evolution

ICSE'16 Companion, May 14–22, 2016, Austin, TX, USA.
© 2016 ACM. ISBN 978-1-4503-4205-6/16/05…$15.00.
DOI: http://dx.doi.org/10.1145/2889160.2889238

1. INTRODUCTION

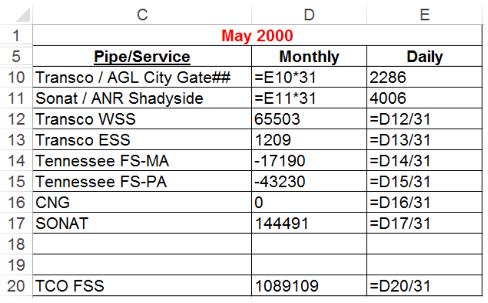
Spreadsheets have been widely used by companies for various business tasks, such as data capturing and analysis, decision support, financial reporting, and so on. Scaffidi [29] estimated that over 55 million users in the United States used spreadsheets in 2012.

Since spreadsheets are mostly written by users unfamiliar with software engineering practice, errors are easily induced into spreadsheets during maintenance and updates [26]. If these errors are not timely detected and fixed, they can induce great financial losses [24]. In order to improve the quality of spreadsheets, researchers have applied many software engineering methods and techniques, which have been developed for conventional programs, on spreadsheets, such as testing [1][12], error detection [7][9][17], and debugging [2][28].

Like conventional program code, spreadsheets can be copied, modified and renamed during evolution. The studies on conventional program evolution have significantly affected software engineering practice. Examples of these studies include clone [20][23][30], defect predication [13][21], and bug fixing [19]. However, there are few studies made on the spreadsheet evolution despite its importance [18]. The unavailability of industrial-scale spreadsheet corpora with change histories and version information is a key obstacle to study spreadsheet evolution problems scientifically. First, the change history of spreadsheets is rarely documented. A lot of spreadsheets are being maintained without version control [10]. Second, although some companies may use SharePoint[1], Google Spreadsheets[2], SpreadGit[3] or other version management tools (e.g., Github[4]) to store the version information of spreadsheets, the information is not publicly accessible due to business confidentiality. Third, the two most popular spreadsheet corpora used by a significant amount of prior work on spreadsheets are EUSES [11] and Enron [14]. However, both of them do not include any version information, and no relations among the spreadsheets are provided. As such, it is difficult to infer the context in which the spreadsheets were created and modified.

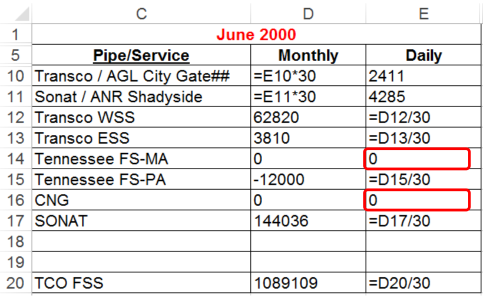
Lack of version information also puts a major threat to the quality assurance of spreadsheets and impose difficulties in tracing root causes of spreadsheet errors [10]. In order to understand the issues arising from spreadsheet evolution, extracting versions across spreadsheets is a practical problem encountered by the industry.

[1] https://products.office.com/en-us/sharepoint
[2] http://www.google.com/sheets
[3] https://spreadgit.com/
[4] https://github.com



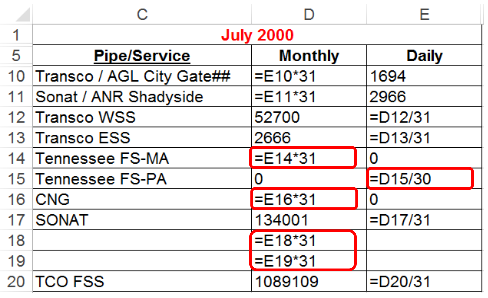
	C	D	E
1	May 2000		
5	Pipe/Service	Monthly	Daily
10	Transco / AGL City Gate##	=E10*31	2286
11	Sonat / ANR Shadyside	=E11*31	4006
12	Transco WSS	65503	=D12/31
13	Transco ESS	1209	=D13/31
14	Tennessee FS-MA	-17190	=D14/31
15	Tennessee FS-PA	-43230	=D15/31
16	CNG	0	=D16/31
17	SONAT	144491	=D17/31
18			
19			
20	TCO FSS	1089109	=D20/31

***v*1. Storage services in May**



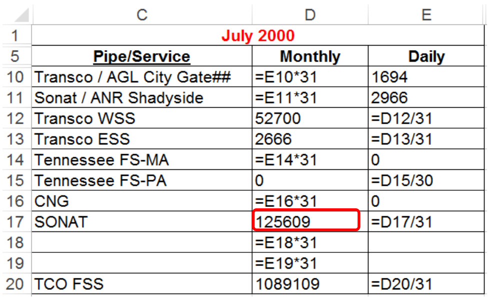
	C	D	E
1	June 2000		
5	Pipe/Service	Monthly	Daily
10	Transco / AGL City Gate##	=E10*30	2411
11	Sonat / ANR Shadyside	=E11*30	4285
12	Transco WSS	62820	=D12/30
13	Transco ESS	3810	=D13/30
14	Tennessee FS-MA	0	0
15	Tennessee FS-PA	-12000	=D15/30
16	CNG	0	0
17	SONAT	144036	=D17/30
18			
19			
20	TCO FSS	1089109	=D20/30

***v*2. Updated for June, removed two formulas (E14 and E16)**



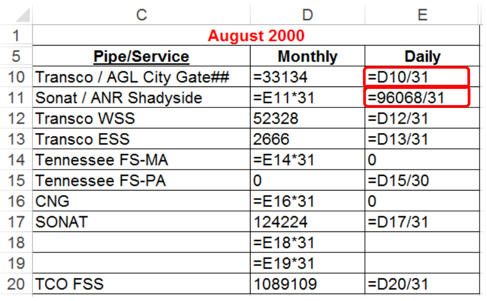
	C	D	E
1	July 2000		
5	Pipe/Service	Monthly	Daily
10	Transco / AGL City Gate##	=E10*31	1694
11	Sonat / ANR Shadyside	=E11*31	2966
12	Transco WSS	52700	=D12/31
13	Transco ESS	2666	=D13/31
14	Tennessee FS-MA	=E14*31	0
15	Tennessee FS-PA	0	=D15/30
16	CNG	=E16*31	0
17	SONAT	134001	=D17/31
18		=E18*31	
19		=E19*31	
20	TCO FSS	1089109	=D20/31

***v*6. Updated for July, added four formulas (in column D) that were previously missed, forgot to adjust one formula (E15)**



	C	D	E
1	July 2000		
5	Pipe/Service	Monthly	Daily
10	Transco / AGL City Gate##	=E10*31	1694
11	Sonat / ANR Shadyside	=E11*31	2966
12	Transco WSS	52700	=D12/31
13	Transco ESS	2666	=D13/31
14	Tennessee FS-MA	=E14*31	0
15	Tennessee FS-PA	0	=D15/30
16	CNG	=E16*31	0
17	SONAT	125609	=D17/31
18		=E18*31	
19		=E19*31	
20	TCO FSS	1089109	=D20/31

***v*7. Updated data (D17) for July**



	C	D	E
1	August 2000		
5	Pipe/Service	Monthly	Daily
10	Transco / AGL City Gate##	=33134	=D10/31
11	Sonat / ANR Shadyside	=E11*31	=96068/31
12	Transco WSS	52328	=D12/31
13	Transco ESS	2666	=D13/31
14	Tennessee FS-MA	=E14*31	0
15	Tennessee FS-PA	0	=D15/30
16	CNG	=E16*31	0
17	SONAT	124224	=D17/31
18		=E18*31	
19		=E19*31	
20	TCO FSS	1089109	=D20/31

***v*9. Updated for August, adjusted two formulas (E10 and E11)**

Figure 1. A real-world evolution group.

However, no methodologies or tools are available to address this problem. In this paper, we recover the missing change history by clustering spreadsheets that are likely the multiple versions originated from the same spreadsheet into an *evolution group*. We build a spreadsheet corpus VEnron with change histories and version information. VEnron was extracted from the email archive within the Enron Corporation [22], which is one of the largest industrial, real-world data set. In the Enron email archive, we observe that, due to the lack of version control systems, users often exchange their new, updated or revised spreadsheets to others by emails. Therefore, the valuable version information about spreadsheets are hidden in these emails. We cluster the same or similar spreadsheets, and use the information contained in the emails and spreadsheets to recover the linkage among spreadsheets, such as the sending time of an email, the filenames and contents of spreadsheets, and so on. After obtaining the versioned corpus, we perform several spreadsheet evolution analyses on it.

VEnron provides the first publicly available spreadsheet corpus with the historical version information. It serves a valuable resource to facilitate future scientific studies on spreadsheet maintenance. The contributions of this paper are as follows:

- A semi-automated approach to identifying spreadsheets that correspond to multiple versions originated from the same spreadsheet and clustering them into evolution groups.
- An industrial-scale and public spreadsheet evolution corpus of 360 evolution groups, including 7,294 spreadsheets (http://sccpu2.cse.ust.hk/venron/).
- An analysis on these evolution groups, including the users involved in a group, spreadsheet changes, error trend.

The remainder of this paper is organized as follows. Section 2 gives a real-world example about spreadsheet evolution. Section 3 proposes the details of our extraction approach. Section 4 presents our analysis on VEnron. We discuss our approach and results in Section 5, and related work in Section 6. Finally, we review our contributions and new research directions in Section 7.

2. A REAL-WORLD EXAMPLE

In this section, we illustrate version information using several related spreadsheets extracted from the Enron email archive [22]. We explain how to recover the versions among these spreadsheets.

2.1 Example

Figure 1 gives five worksheets whose spreadsheet files are clustered into an evolution group. They report the monthly and daily amount of different storage services in each month. Since each month has different numbers of days, the constants used in the formulas vary across months. For example, in version *v*1, the formulas (e.g., the one in D10) use 31 (the number of days in May) as constants, and in version *v*2, the formulas (e.g., the one in D10) use 30 (the number of days in June) as constants. These five worksheet names and their associated spreadsheet filenames are listed Table 1. We found 11 spreadsheets in this evolution group. Since the related worksheets (FOM Storage) in versions *v*3, *v*4, *v*5, *v*8, *v*10 and *v*11 are the same as their previous versions, we do not show them in Figure 1.

2.2 Spreadsheet Evolution in the Example

In Figure 1, we use the red rounded rectangles to show the key changes from its preceding version. We observed several interesting changes.

We did not find any inconsistency among the formulas in version *v*1. When version *v*1 evolves to *v*2, the formulas in cells E14 and E16 are changed to 0. These values are consistent with those computed by the original formulas in *v*1 because cells D14 and D16 contain 0. It is likely that users entered a 0 value to E14 and E16 before doing so for D14 and D16. This is because users do not need to substitute the original formulas in E14 and E16 with 0 if they first made changes to D14 and D16. We note that *v*2 induces a discrepancy on how values are computed in cells E12:E17.

In version *v*6, users likely found that the values in cells D14 and D16 should be computed instead of a constant value 0. They added formulas in cells D14 and D16. The users also introduced another two issues. (1) The formula in cell E15 should be D15/31 (31 days in July). Since cell D15's value is 0, no data error happens. (2) The users added two formulas to cells D18 and D19. The rows 18 and 19 are empty, and are not used in the spreadsheets. In the later version *v*7, the users changed the value of cell D17, but they did not notice the issue in cell E15.

In version *v*9, users changed the formulas in cells D10 and E10. Cell E11's formula was changed to 96068/31 (where its input cell D11's value is 96068). This is likely an error. Although Excel gives a warning at cell E11 (its formula is inconsistent with its neighboring cells), the users did not fix it. We find that the error in cell E15 remains in the subsequent versions (*v*10 and *v*11).

According to the Enron email archive [22], two users were involved in the modifications of these spreadsheets. We found that more errors were introduced during evolution, and some errors in these spreadsheets can last for several versions.

2.3 Version Information in Spreadsheets

From the evolution group in Figure 1, we find that the hidden version information in the spreadsheets can be manifested in several ways. (1) The spreadsheet filenames may suggest the files of different versions. In Table 1, all the spreadsheets follow the naming convention "<Month>00_FOM_Req<id>.xsl", which indicates these spreadsheets are likely the different versions of "FOM_Req.xsl". Here, <Month> can be May/Jun/Jul(y)/Aug, and <id> is an integer index referring to the same month. (2) The worksheet names can suggest the different versions, too. For example, all the five worksheets follow the naming convention "FOM <Month> Storage". This indicates that these worksheets are likely the different versions of "FOM Storage". (3) The worksheet contents may indicate their versions. In one case, the title of each worksheet shows the month of the worksheet. In another case, all the tables in the five worksheets show similar structures. This indicates that they should be evolved from the same original spreadsheet.

The version information in the spreadsheets can also be used to determine their version order. For example, based on the temporal order, we can infer that version *v*2 should follow version *v*1.

2.4 Version Information in Emails

When users sent a spreadsheet by email, they often explained in the same email the changes that they made to the spreadsheet. For example, the user in an email containing the spreadsheet Jun00_FOM_Req.xls (*v*2) said: "*The attached file is an update to the original one sent on Friday with our May daily volume requirements. Please refer to the 'comments' worksheet and the comments dated 4/24/00 for differences between this version and the original*". The other user in his email said: "*The attached file contains updated July 2000 volume requirements for CES. The changes relative to the original request sent on 6/23/00. Changes are described on the worksheet labelled 'Comments' under the date 06/26/00*".

Table 1. The spreadsheets and worksheets in our motivating evolution group

Version id	Spreadsheet filename	Worksheet name
*v*1	May00_FOM_Req2.xls	FOM May Storage
*v*2	Jun00_FOM_Req.xls	FOM Jun Storage
*v*6	July00_FOM_Req.xls	FOM Jul Storage
*v*7	July00_FOM_Req02.xls	FOM Jul Storage
*v*9	Aug00_FOM_Req.xls	FOM Aug Storage

This indicates that describing spreadsheet changes in emails is commonly adopted by users. The information in these emails is useful to confirm the relationship between different spreadsheets.

2.5 Approach Overview

Two technical challenges need to be addressed when building a versioned spreadsheet corpus on the Enron email archive [22]. The first challenge is how to check if a spreadsheet is similar to another one. Unlike conventional software, spreadsheets are a special kind of programs, which are indexed by two-dimensional cell addresses. Code clone detection approaches on conventional software [5] cannot be used to measure the similarity of spreadsheets. Data clone detection in spreadsheets [17] can only detect a region of cells with the (almost) same data, and cannot measure the similarity of spreadsheets, because different versions (e.g., versions *v*1 and *v*2 in Figure 1) often have different data. Therefore, no existing tools can help us measure the similarity of contents in spreadsheets. Worse, there are many (15,879) spreadsheets in the Enron email archive. It is impractical to check each pair of spreadsheets one by one. The second challenge is how to decide the order of versioned spreadsheets in a group. The order of versions is implicit in the Enron email archive.

Our approach works as follows: 1) Extract a shortened filename from each spreadsheet by deleting version-related substrings from its filename, such as "May", "Jun", "July", "Aug", "00", "02", "2", and so on in Table 1. 2) Cluster spreadsheets into groups based on their shortened filenames. 3) Identify from each group those spreadsheets that likely belong to the different versions of the same spreadsheet. 4) Determine a version order in each group according to the version information manifested in spreadsheets and emails.

3. BUILDING VENRON

Let us illustrate our approach by building VEnron. We observed that the spreadsheets in the Enron email archive mostly follow the same naming convention varied with version number, such as versions *v*1, *v*2, *v*6, *v*7 and *v*9 in Table 1. We built the VEnron corpus in four steps: 1) Extracted the spreadsheets and their shortened filenames from email files. The extraction process needs to preserve the spreadsheet filenames. 2) Clustered these spreadsheets into different evolution groups based on their shortened filenames extracted. 3) Validated if the spreadsheets in each group share similar worksheet names, table structures (including table titles, row/column labels and cell formulas) and email contents. This step was carried out manually to identify irrelevant spreadsheets in each group. 4) Reordered the spreadsheets in every group according to the version information of these spreadsheets and related emails.

3.1 Obtaining the Enron Email Archive

The spreadsheets extracted by Hermans [14] have discarded the spreadsheets' contexts and version information. Therefore, we used the original Enron email archive as our study subject. We obtained the Enron email archive from its website[5]. The version for the Enron email archive is v1.3, and the update time is 29 July, 2013. The emails in the Enron archive span a period of about 15 months, from August 2000 to December 2001. The Enron email archive has 130 folders (one folder per user), which contain 190 .pst files. These .pst files contain 752,604 .eml files (a .eml file per email) in total. We filtered out the "Contacts" and "Drafts" folders of the mailboxes in our study.

3.2 Extracting Spreadsheets and Related Emails

To extract spreadsheets and related emails, we used JavaMail[6], which is a platform-independent and protocol-independent framework to build mail and messaging applications. We identified 41,945 related emails that contain at least one spreadsheet as attachment. These emails contain altogether 49,863 spreadsheets. If two spreadsheets have the same filename and MD5 file hash, we assumed that they are the same spreadsheet, and kept only one of them. If two spreadsheets have the same MD5 file hash but different filenames, we kept both of them. Finally, we extracted 17,152 unique spreadsheets from these emails. After removing those spreadsheets that are password-protected, saved in very old Excel format or contain damaged worksheets, we obtained 15,879 spreadsheets. We also kept the email content associated with these spreadsheets.

3.3 Clustering Spreadsheets

It is labor-intensive to manually cluster 15,879 spreadsheets into different evolution groups, in which all the spreadsheets have similar/same structures and semantics. Hence, we partially automate the process by clustering spreadsheets according to their filenames. The process is motivated by an observation made from the Enron spreadsheets: *the difference between the similar filenames occurs often only in numbers, date time, or special characters (e.g., "final" and "_")*. An example is given in Table 1. The observation enables us to cluster spreadsheets based on their filenames shortened by removing three following types of characters.

- ***Numbers:*** Numbers can be used as version IDs, months, years, and so on. For example, for the spreadsheet filename July00_FOM_Req02.xls in Table 1, "00" is used as year 2000, and "02" is used as the version ID.

- ***The full and abbreviated month names:*** The months can be used as the different versions in an evolution group. For example, for the spreadsheet filename July00_FOM_Req02.xls in Table 1, "July" is used as the version of July's spreadsheet. We delete january, jan, february, feb, march, mar, april, apr, may, june, jun, july, jul, august, aug, september, sep, october, oct, november, nov, december and dec.

- ***Some special characters and words:*** Special characters, such as _, - , (,), ~, +, $, and #, can be used to combine different parts of the filenames. The special words "version" and "final" can be used to mark versions. The spreadsheet filenames' suffix ".xls" is the same for all the spreadsheets in the Enron email archive, so we delete it, too.

[5] http://info.nuix.com/Enron.html

[6] http://javamail.java.net

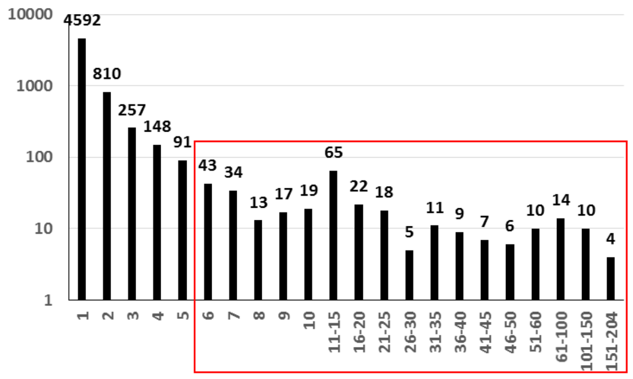


Figure 2. Identified evolution groups in the Enron email archive (horizontal axis: the number of spreadsheets in a group, vertical axis: the number of groups, the groups in the rectangular box have been manually validated).

Note that we kept the spreadsheets with the same MD5 file hashes but different in filenames. Take the Enron spreadsheets "NP 2-26.xls" and "NP 15 pages.xls" as examples. They have the same MD5 hash values. After deleting the special characters from their filenames, we got "NP" and "NPpages". We found that there are 70 spreadsheets in the group "NPpages", and only 1 spreadsheet in the group "NP". Since the chances of extracting useful version information from a group of one spreadsheet are low, we excluded the "NP" group from VEnron.

We found that the filenames of 404 spreadsheets are shortened to an empty string after removing the three types of characters. These spreadsheets are excluded from VEnron because it is hard to cluster them based on an empty string. Figure 2 shows the identified evolution groups from the Enron email archive. In practice, it is unnecessary to validate all 6,205 groups (15,475 spreadsheets) because those groups that contain too few spreadsheets unlikely expose issues arising from spreadsheet evolution. As such, we manually validated only those groups that contain more than five spreadsheets. This contributes to 307 groups and 46.9% (7,445/15,879) of the spreadsheets in the Enron email archive.

3.4 Validating the Evolution Groups

As explained in Section 3.3, evolution groups are clustered by means of heuristics over spreadsheet filenames. The clustering is subject to three types of errors. 1) Clustered groups may contain unrelated spreadsheets where no version information can be found. 2) Multiple versions of the same spreadsheet may be clustered into multiple groups. 3) The versions of different spreadsheets may be clustered into the same group. To address this, we need to manually validate the spreadsheets in the clustered groups.

The key idea to validate a group is to check whether all the spreadsheets in a group share similar table structures and formulas. If we cannot find such similarity in a spreadsheet, we remove it from the group. For each spreadsheet that is removed from a group, we further check if it belongs to another group. We delete groups that contain one spreadsheet after the validation process.

Since we are not the authors of the Enron spreadsheets, we employed the Spreadsheet Compare tool (a Microsoft Excel 2013 Add-In) [31] to compare two spreadsheets. Spreadsheet Compare can compare two spreadsheets (or two versions of the same

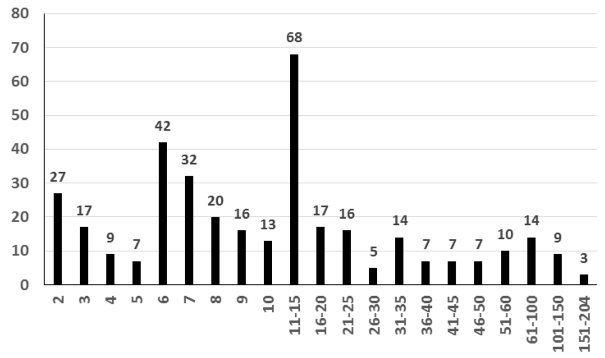


Figure 3. Confirmed evolution groups in Enron (horizontal axis: the number of spreadsheets in a group, vertical axis: the number of groups).

spreadsheets), and shows the differences by means of histograms and colors.

We use the following heuristics to determine whether two spreadsheets should belong to the same group:

- ***Similarity on spreadsheet filenames:*** Two spreadsheets likely belong to the same group if their filenames share a meaningful substring. The shared substring often briefly explains the purpose of the spreadsheets. An example is the substring "FOM_Req" in Table 1.
- ***Similarity on worksheet names:*** Two spreadsheets likely belong to the same group if each contains more than one worksheet name that matches a worksheet name found in another spreadsheet. An example is the string "FOM Storage" of the worksheet names in Table 1. As a special case, the default names of worksheets (e.g., Sheet1 and Sheet2) are not considered.
- ***Similarity on the structure of corresponding worksheets:*** At least two worksheets, which come from two spreadsheets respectively, have similar layout. For example, the titles (labels) of tables and formulas are similar. Therefore, these worksheets may have similar semantics.
- ***Email contents:*** Email contents may tell if a spreadsheet was an update to another one. This enables us to confirm if two spreadsheets belong to the same group.

Figure 3 shows the results of our manual validation. We confirmed 360 groups, which contain 7,294 spreadsheets (45.9% of 15,879 spreadsheets in total), from the 307 clustered evolution groups. We find that our clustering approach in Section 3.3 precisely identified 231 (64.2%) out of the 360 groups. This suggests that our hypothesis is reasonable.

3.5 Recovering Version Orders

The version order of spreadsheets that belong to the same group provides critical information of update history. Given a group of spreadsheets, it is challenging to recover the version order among them. This is because we do not know how the spreadsheets were created or modified. The version orders can be complex in the Enron email archive. However, we find that the version orders of our 360 confirmed groups can be represented by a total order. In the following, we explain the heuristics based on which a total order for each of these evolution groups can be recovered.

Several pieces of information extracted from the Enron email archive can help us figure out the version orders. 1) Version order information can be manifested in the spreadsheet filenames, worksheet names, and the contents in the spreadsheets. For example, in Table 1, the spreadsheet filenames and worksheet names can contain the version orders. 2) Emails have timestamps. The emails' sending time may reflect the version orders of attached spreadsheets. For example, an update version can be sent later after the original one. 3) The email contents may also reflect from which spreadsheet the current one was derived. We describe these heuristics as follows.

Table 2. The statistics of the groups containing time

The type of time	The number of groups
Spreadsheet filenames	200
Worksheet names	59
Worksheet contents	278
None	43

3.5.1 The Time in the Spreadsheets/Worksheets

The version orders can sometimes be inferred from the spreadsheet artefacts such as filenames, worksheet names, and worksheet contents. The date/time and index information found in these artefacts provide useful hints on the time when a spreadsheet was created/modified.

Table 2 shows the statistics of the spreadsheets in the 360 groups that contain artefacts encoding time information. We note that only 11.9% (43/360) groups contain spreadsheets whose artefacts encode no time information. 88.1% (317/360) groups encode at least one piece of time information in their spreadsheet artefacts. We also find that 58.3% (210/360) groups encode at least two pieces of time information in their spreadsheet artefacts. The encoded time information in these groups can help identify the version orders of spreadsheets.

Spreadsheet filenames: The spreadsheet filenames often contain various version information. For example, in Table 1, all the spreadsheet files are named using the convention "<Month>00_FOM_Req<id>.xsl", where <Month> is May/Jun/July/Aug, <id> is the spreadsheet index for the same month. From these filenames, we can infer the version order. As shown in Table 2, 200 (55.6%) out of 360 groups have spreadsheets encoding time information in their filenames.

Worksheet names: Worksheets in a spreadsheet may be named by a certain date/time. For example, in Table 1, the worksheets are named using the convention "FOM <Month> Storage", where <Month> is May/Jun/Jul/Aug. In another case, worksheets are named by the date when they were created or updated. One spreadsheet may contain more than one worksheet of this type. In the case where a spreadsheet contains multiple worksheets named by dates, we associate the latest one among these dates with the spreadsheet. Spreadsheets that encode time information in their embedded worksheet names can be found in 59 (16.4%) out of 360 groups as shown in Table 2.

The time in the worksheet contents: Cells in worksheets may contain the time when the worksheets were created/modified. For example, in Figure 1, the first row in each worksheet shows this case. This type of cells usually appear on the left or top of a table, or serve as the labels of a table. We observe that some spreadsheets in a group may add or modify records daily, monthly or yearly. If there is more than one cell of this type, then we select the latest time as the creation/modification time of the spreadsheet. As shown in Table 2, 278 (77.2%) out of 360 groups encode time information in this way.

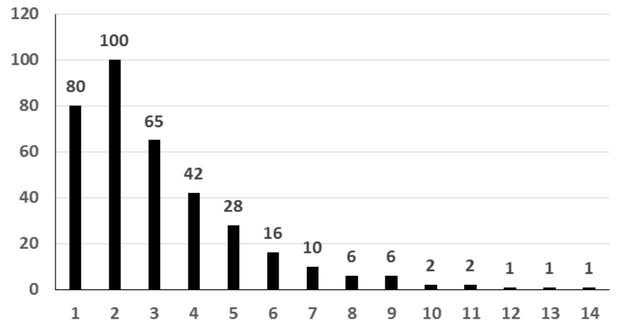


Figure 4. The distribution of committers in VEnron (horizontal axis: the number of committers involved, vertical axis: the number of groups).

3.5.2 The Email Sending History

In the Enron email archive, all the spreadsheets were exchanged through email attachments. The emails' sending time can suggest when the spreadsheets were created or modified. For example, we assume that *S'* is an updated version of *S* if they belong to the same group and *S'* was sent by an email later than that sending *S*. In the case where a spreadsheet was exchanged by multiple emails, we recover spreadsheet version order based on the earliest email.

Given two different spreadsheets *S* and *S'* in a group. In the following two cases, we consider that *S'* is a later version of *S*.

- A user sends *S* before *S'* to another user. In this case, the first user might have updated *S*, saved it as *S'*, and then sent *S'* to the second user.
- A user receives *S* before sending *S'* by email. In this case, the user might have modified *S*, saved it as *S'*, and then sent *S'* by email.

Note that a user may send the newer version *S'* before the older version *S*. We can use the heuristics in Sections 3.5.1 and 3.5.3 to identify and rectify such scenarios.

3.5.3 Email Contents

When users email a spreadsheet, they may put down in the email some description of the spreadsheet as well as the changes they have made. For example, "*The attached file is an update to* ***the original one sent on Friday*** *with our May daily volume requirements*", and "*…the changes relative to* ***the original request sent on 6/23/00****…*". The description facilitates us to locate an earlier version of the spreadsheets attached in the email.

4. EVOLUTION ANALYSIS ON VENRON

It is interesting to study how spreadsheets evolve in companies, and specially, whether VEnron contains interesting changes for further studies. We perform several analyses to study the evolution characteristics of VEnron.

4.1 Basic Statistics of VEnron

Table 3 shows an overview of VEnron. 251 (69.7%) of the 360 groups contain formulas. In total, 7,294 spreadsheets (4,149 containing formulas) are analyzed, containing 35,373 worksheets, which is an average of 4.8 worksheets per spreadsheet. These spreadsheets have 2,841,073 rows and 500,821 columns respectively. These spreadsheets together have 41,208,082 non-empty cells of which 9,225,740 have formulas. This is an average of 5,650 cells and 1,265 formulas per spreadsheet.

Table 3. An overview of VEnron

Number of groups	360
Number of groups with formulas	251
Number of spreadsheets	7,294
Number of spreadsheets with formulas	4,149
Number of worksheets	35,373
Number of rows	2,841,073
Number of columns	500,821
Number of non-empty cells	41,208,082
Number of formulas	9,225,740

Table 4. The types of version representations

The type of version representation	# Group
1. Time	240
2. Prefix or suffix	68
3. None	142
At least two of 1, 2, 3	66
All of 1, 2, 3	12

The distribution of evolution groups in VEnron is shown in Figure 3. 183 (50.8%) groups contain no more than 10 spreadsheets, and 165 (45.9%) groups contain 11-100 spreadsheets. 12 (3.3%) groups contain more than 100 spreadsheets. The largest group contains 204 spreadsheets.

4.2 Committers of Spreadsheets

The Enron email archive contains 130 users' email data. If a user sent a spreadsheet different from its previous version, we consider the user a committer of the corresponding group. The committers often gave a new version of a spreadsheet to others.

Figure 4 shows the distribution of committers in the 360 groups. 260 (72.2%) of the evolution groups involve more than one committer. This suggests that spreadsheets were often maintained by multiple users, and consistent modification among users would be important to the integrity of spreadsheets. We find 315 (=80+100+65+42+28), i.e., 87.5% of the groups involve no more than 5 committers (3.8% of all users). This indicates that spreadsheets were often updated by a small group of users, such as the users who belong to the same department or have some business relationship. We find that only 5 (1.4%) of the groups have more than 10 committers.

4.3 Various Version Representations

We find no evidence in the Enron email archive that the spreadsheets were managed by a version control system. This suggests that the spreadsheets were likely stored in their users' local file systems. To avoid conflicting filenames, users deployed different filenames to denote the multiple versions of the same spreadsheet.

To study how users managed the different versions of the same spreadsheets, we manually validated the 360 groups. Three common types of version representations (as shown in Table 4) were found. In Table 4, if a group has used one type of representation in some spreadsheets, we count it as 1. Therefore, a group may use more than one type of version representations. 1) Time is the most common type (66.7%, 240/360) used to represent different versions. For example, in Table 1, users used the months (May, Jun, July, or Aug) to distinct different versions. This result is also reasonable. It is a useful and simple way to manage a set of spreadsheets for users who only care about the freshest data. Users can quickly get the version information about whether a spreadsheet contains more updated data than the other one. 2) Users may use indexes in the filenames' prefixes or suffixes

(18.9%, 68/360) for different versions. For example, in Table 1, version *v*7 "July00_FOM_Req02.xls" uses the suffix "02" to distinct from version *v*6 "July00_FOM_Req.xls". 3) Users did not use any version representation above to distinct different versioned spreadsheets (39.4%, 142/306). For example, the spreadsheets used the same filenames. We find that this case is common in the Enron email archive. Users may not be able to correctly identify different versions in an evolution group.

In Table 4, we also find that users may use different version representations in the same group. For example, in Table 1, versions *v*1 and *v*7 use time and suffix as version presentations in the same time. Among 360 groups, we find that 18.3% (66/360) groups uses at least two types of version representations, and 3.3% (12/360) groups uses all the three version representations. Varied version representation would make it complicated to manage different versions of spreadsheets in a group.

Further, we find duplicated spreadsheet filenames in 71.1% (256/360) groups. Without any further information, users cannot distinct these different spreadsheets. This aggravates the difficulties in applying version management to spreadsheets.

4.4 Spreadsheet Changes during Evolution

In this section, we investigate what changes happened during the evolution in each evolution group.

We employed the tool Spreadsheet Compare 2013 [31] to compare every two adjacent versions in a group. This tool offers 14 options to analyze different types of changes between two spreadsheets. Among these options, four (starting with *SysGen*) are related to the changes generated by Excel. *SysGen* (System Generated) changes are those that are not made directly by users but made by the system (Excel) as a by-product of a user change. For example, when a row is deleted, Excel will automatically change the formula to account for the row deletion by changing the references to cells. Therefore, we did not consider these four *SysGen* changes. Six options are related to cell format, sheet visibility, cell protection, and so on. These changes are unrelated to the correctness of spreadsheets, and we did not consider these changes, too. We are specifically concerned with the following four options, which are related to the correctness of spreadsheets:

- ***Structural:*** The tool can analyze the changes of the structure of each worksheet, including eight different types of operations: *added/ renamed/ deleted sheet at position n*, *moved sheet from position n to position m*, *added/ deleted column(s)* and *added/ deleted row(s)*.
- ***Entered Values:*** The tool can analyze the changes of the value of each cell, including three different types of operations: *entered value added/changed/deleted*.
- ***Formulas:*** The tool can analyze the changes of each formula, including three different types of operations: *formula added/ changed/ deleted*.
- ***Calculated Values:*** The tool can analyze the changes of the value in a formula cell. Those changes are caused when a user changes some values of a formula's input cells. These changes include three different types of operations: *calculated value added/ changed/ deleted*.

We exported the comparison results of Spreadsheet Compare 2013 to spreadsheets, and obtained statistics from these results. The results are shown in Table 5. Calculated value changes are not direct changes made by users, so we ignore them in the following discussion.

Table 5. The statistics of various changes during evolution

The type of change	Total	# Groups	# Spreadsheets
Change on worksheet	3,297	166	1,357
Change on row	332,957	275	2,802
Change on column	17,451	204	1,518
Change on entered value	8,347,649	346	6,912
Change on formula	1,755,900	217	2,860
Change on calculated value	2,794,083	234	3,817
Other changes	1,087,420	284	3,674

4.4.1 Structural Changes

Worksheet changes: Among the 360 groups, 166 groups contain 3,297 structural worksheet changes. The changes concern 9.3% of all worksheets and spread across 1,357 (18.6%) of all spreadsheets.

Row/column changes: Among the 360 groups, 275 groups contain 332,957 structural row changes and 204 groups contain 17,451 structural column changes. These row and column changes concern 11.7% of all rows and 3.5% of all columns, respectively. The row changes spread across 2,802 (38.4%) of all spreadsheets while the column changes spread across 1,518 (21.9%) of all spreadsheets.

4.4.2 Entered Value Changes

Among the 360 groups, 346 groups have entered value changes. In total, 6,912 (94.8%) spreadsheets contain 8,347,649 entered value changes, which concern 20.3% of all cells. This indicates that most spreadsheets contain entered values changes. Intuitively, structural changes are relatively fewer than cell value changes as spreadsheets evolve. For example, users make cell value changes when they update the corresponding cells in last month's report to generate a similar report for this month (e.g., version *v*9 in Figure 1).

4.4.3 Formula Changes

Among the 360 groups, 217 groups have formula changes. In total, 2,860 (39.4%) spreadsheets contain 1,755,900 formula changes, which concern 19.0% of all formula cells and 4.2% of all cells.

The changes in formula cells could introduce errors. For example, in version *v*2 of Figure 1, the formulas in cells E14 and E16 were wrongly removed; in version *v*6, the formula in cell E15 was not consistently updated. In the future, we will investigate these formula changes, and try to identify and validate errors in them.

From above spreadsheet change analyses during evolution, we can conclude: 1) For structural changes, row changes are much more common than column changes (11.7% vs. 3.5%). 2) For cell changes, entered value changes are much more common than formula changes (20.3% vs. 4.2%). 3) Formula changes are common (19.0% of all formula cells), too.

4.5 The Error Trend

We do not have the ground truth telling the intended semantics of formulas in the spreadsheets of VEnron. It is therefore difficult for us to decide which cells are erroneous. As such, we followed a recent work by Hermans [14] and analyzed those errors reported by Excel. Seven types of errors were reported: 1) divide by 0 (#DIV/0!), 2) a formula or a function inside a formula cannot find its referenced data (#N/A!), 3) the text in a formula is not recognized (#NAME?), 4) a space was used in a formula that references multiple ranges (#NULL!), 5) a formula has invalid numeric data (#NUM!), 6) a reference is invalid (#REF!), 7) the wrong type of operand or function argument is used (#VALUE!).

To get insights into the error trend during evolution, we analyzed the number of formulas that result in Excel errors in every spreadsheet in each evolution group. Excel reported at least one error in 72 (20.0%) of the 360 groups. In the following discussion, we only consider these 72 groups. Figure 5 shows the cumulative results of added and removed Excel errors for each group during evolution. We can see that Excel errors change a lot during evolution.

In order to analyze the trends of Excel errors in an evolution group, we built a linear regression model $y = mx + b$ by the least square method (LSM), where x is the version ID (starting from 1), and y is the number of Excel errors in the spreadsheet with version ID x. Therefore, if the slope $m > 0$, it indicates that the number of errors increases during evolution, whereas, if the slope $m < 0$, it indicates the number of errors decreases during evolution. Figure 6 shows how the slopes change in these 72 groups. The numbers of errors increase in 42 groups; the numbers do not change in 7 groups, and the numbers decrease in 23 groups.

We also studied whether new errors were introduced during evolution. For two adjacent versions in an evolution group, if the later version contains more errors than the previous version, we consider that the later version introduced new errors. We find that 61 (16.9%) of all groups introduced new errors.

4.6 The Emails Describing Spreadsheets

In order to collect further information about the versioned spreadsheets in VEnron, we analyzed their concerning emails.

The 7,294 VEnron spreadsheets concern 8,587 non-redundant emails. Following the email analysis in a recent work by Hermans [14], we indexed the emails based on a set of keywords found in their contents. Spreadsheets are mentioned in 648 (7.5%) emails, which contain one of the following keywords: model, spreadsheet, excel or worksheet. Spreadsheet errors are described in 468 (5.5%) emails, which contain one of the following keywords: error, mistake, problem, discrepancy, anomaly, anomalies, incorrect, bug, fault or failure. Spreadsheet modification are described in 1,347 (16.7%) emails, which contain one of the following keywords: new version, update, change, revision, revising or revised. Examples of such emails can be found in Section 2.4.

We find altogether 1,948 (22.7%) emails describing spreadsheets, errors or modifications. Still, 6,639 (77.3%) emails do not refer spreadsheets in their contents. These emails provide no information on the changes made to their attached spreadsheets.

5. DISCUSSION

In prior sections, we have introduced a semi-automated approach to recover version information of spreadsheets and applied it to the Enron email archive, constructing a versioned spreadsheet corpus VEnron. The approach is based on several heuristics (such as the spreadsheets in an evolution group have the similar filenames) and manual validation. In this section, we discuss a variety of issues that may affect the applicability and suitability of our proposed approach and VEnron.

5.1 Enron Email Archive

1) Incomplete Enron email archive: To protect user privacy, all the sensitive personal information in the Enron email archive had been removed before being publicly available [6]. This cleaning process may have removed some version information.

2) Spreadsheets saved in unsupported Excel formats or with password protection: Some spreadsheets in the email archive are

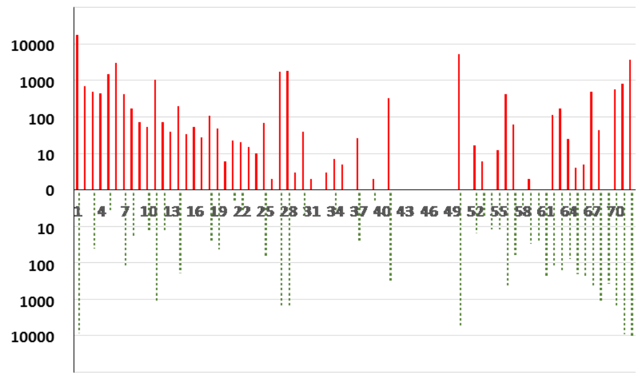


Figure 5. The added and removed errors in 72 groups (horizontal axis: the group id, vertical axis: the cumulative number of Excel errors, red and solid lines denote the cumulative number of added Excel errors, green and dashed lines denote the cumulative number of removed Excel errors).

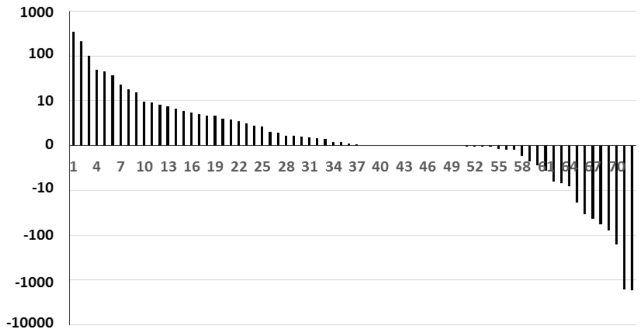


Figure 6. The error trends in 72 groups (horizontal axis: the group id, vertical axis: the slope of linear regression result).

password-protected, or contain damaged worksheets. We omitted them from VEnron. Our employed spreadsheet analysis tool Java Excel API [32] cannot handle the spreadsheets saved in very old unsupported Excel formats. 1,273 spreadsheets fall into this category. Some of these spreadsheets may carry version information.

5.2 Undetected Spreadsheet Versions

1) Dissimilar spreadsheet filenames: Our spreadsheet clustering approach is based on an assumption that the spreadsheets in a group share similar filenames. This would exclude the spreadsheets with dissimilar filenames in a group.

2) Missing evolution groups: Although 8,685 spreadsheets were not clustered by our approach into any evolution groups, it is possible that some of these spreadsheets contain version information and form evolution groups. Since there are numerous possible ways to partition these spreadsheets, clustering them manually is impractical. A tool, which can measure the similarity of spreadsheet contents, may help identify these missing evolution groups, and alleviate this threat.

5.3 Evolution Orders

1) Parallel evolution: Currently, we find that total order works well for our identified evolution groups. In practice, this observation may not always hold. A user may create two different successive versions from the same spreadsheet. Our approach cannot capture this type of spreadsheet evolution. The reason is that it is difficult to extract this evolution from the email and spreadsheet contents.

2) Lost evolution: Without the support of version management systems, users may make a lot of small changes in spreadsheets before exchanging them in emails. Some interesting evolution with respect to these small changes could have been lost.

5.4 Manual Validation

One threat to internal validity of our approach is that we were unable to validate analysis results of spreadsheet evolution in the VEnron corpus by their original users. Therefore, we validated the results by ourselves manually in due diligence. For example, we consider various information from emails and spreadsheets to judge the evolution orders. To alleviate this threat, two authors of this paper have cross-checked all the results.

6. RELATED WORK

In this section, we present and discuss related work in recent years. We focus on those pieces of work that concern spreadsheet corpora, spreadsheet evolution and spreadsheet error detection.

Spreadsheet corpora. EUSES [11] is a spreadsheet corpus widely used by the software engineering community in studying spreadsheets. The EUSES corpus contains 4,037 spreadsheets published in 2005. The spreadsheets of the EUSES corpus were extracted from World Wide Web. These spreadsheets are not necessarily used by real companies. The EUSES corpus has been for problem motivation and empirical experimentation by most existing work including GoalDebug [2], data clone detection [17], AmCheck [9] and CUSTODES [8]. In 2015, Hermans [14] published another spreadsheet corpus based on the over 15,000 spreadsheets found in Enron email archive. This is the first corpus built entirely on top of spreadsheets used by a real company. FUSE [4] is the biggest corpus (containing about 250,000 spreadsheets) so far. FUSE was extracted from a public web archive with about 26 billion web pages. However, EUSES, Enron and FUSE corpora do not provide any version information. All the spreadsheets in them are archived independently. Unlike these three corpora, VEnron provides a versioned spreadsheet corpus, facilitating studies, e.g., evolution analysis and regression testing, which require version information.

Spreadsheet evolution. We only notice one piece of work [18] related to spreadsheet evolution, which compared 54 pairs of spreadsheets consisting of the original spreadsheets developed by the customers and the rebuilt ones created by a professional company F1F9. But due to the commercial confidentiality, these spreadsheets are not publicly available. VEnron can complement this work by studying the evolution process through the 360 evolution groups.

Spreadsheet error detection. Spreadsheet errors are common [25][27]. Therefore, various techniques have been proposed to detect spreadsheet errors. UCheck [3] used the type system to check unit errors. Hermans et al. proposed to visualize spreadsheets by dataflow graphs [15], and detect inter-worksheet smells in these graphs [16]. Our work AmCheck [9] and CUSTODES [8] detect ambiguous computation smells and model spreadsheet smells as outliers, respectively. For these studies, it is difficult to validate the detected errors and smells. VEnron makes it possible to validate the results by cross-checking different versions of the same spreadsheet.

7. CONCLUSION AND FUTURE WORK

7.1 Conclusion

In this paper, we introduce a versioned spreadsheet evolution corpus VEnron, which was extracted from the industrial Enron email archive. VEnron comprises 7,294 spreadsheets in 360 groups and their evolution relationship. Multiple versions of the same spreadsheet are grouped together. We have performed a preliminary analysis on all the evolution groups in VEnron, and found interesting results. To the best of our knowledge, VEnron is the first spreadsheet corpus with version information provided. Our contributions are twofold:

- An industrial-scale and public spreadsheet evolution corpus, containing 360 evolution groups and 7,294 spreadsheets.
- Spreadsheet evolution analyses on these industrial evolution groups, including committers, version representations, error trends, and so on.

The analysis results suggest that VEnron contains version information reflecting real-world changes for further studies, specially:

- Most (87.5%) evolution groups have less than 5 committers.
- Users deploy various ways to represent version information of spreadsheets. Users may adopt more than one representation across multiple versions.
- Formula changes are common (19.0% of all formula cells) during evolution.
- New errors can be easily (61 groups; 16.9% of all groups) introduced during evolution.

7.2 Future Work

We have published VEnron corpus, and believe that this paper lays the foundation for more evolution analysis on spreadsheets. We have identified several interesting research directions.

1) Mining more evolution groups from the Enron email archive: Although we have found lots of evolution groups and spreadsheets with version information, the version information for about half of the spreadsheets in the email archive are still unclear. We need to develop a new approach to handle these spreadsheets.

2) The error detection and fixing in spreadsheets: After we obtain the historical information of an evolution group, we can use them to detect the inconsistencies between different versions, and fix them by combining different semantics from different versions.

3) Study on the semantic error evolution of spreadsheets: It is interesting to know how a semantic error (e.g., cell E15 in *v*6 and *v*9 of Figure 1) was introduced into a spreadsheet, how and when it was fixed during evolution.

4) Designing a version control software for spreadsheets: Our analysis has shown that users may use different types of version representations for their intentions. As we have shown in Table 1, we can use months as versions to represent the reports of different months, and indexes for the versions in a month. For a version control system for spreadsheets, it should provide the flexibility for different kinds of version representations.

8. ACKNOWLEDGMENTS

This work was supported in part by National Grand Fundamental Research 973 Program (2015CB352201), and by the Hong Kong SAR RGC/GRF grant (611811), and by Beijing Natural Science Foundation (4164104).

9. REFERENCES

[1] R. Abraham and M. Erwig. AutoTest: A Tool for Automatic Test Case Generation in Spreadsheets. In *IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, pages 43 –50. 2006.

[2] R. Abraham and M. Erwig. GoalDebug: A Spreadsheet Debugger for End Users. In *Proceedings of the 29th International Conference on Software Engineering (ICSE)*, pages 251–260. 2007.

[3] R. Abraham and M. Erwig. UCheck: A Spreadsheet Type Checker for End Users. *Journal of Visual Languages & Computing*, 18(1):71–95, 2007.

[4] T. Barik, K. Lubick, J. Smith, J. Slankas, and E. Murphy-Hill. FUSE: A Reproducible, Extendable, Internet-scale Corpus of Spreadsheets. In *Proceedings of the 12th Working Conference on Mining Software Repositories (MSR)*. 2015.

[5] S. Bellon, R. Koschke, G. Antoniol, J. Krinke, and E. Merlo. Comparison and Evaluation of Clone Detection Tools. *IEEE Transactions on Software Engineering (TSE)*, 33(9):577–591, 2007.

[6] A. Cassidy and M. Westwood-Hill. Removing pii from the edrm enron data set: Investigating the prevalence of unsecured financial, health and personally identifiable information in corporate data. [Online]. Available: http://www.nuix.com/images/resources/case_study_nuix_edrm_enron_data_set.pdf.

[7] C. Chambers and M. Erwig. Reasoning About Spreadsheets with Labels and Dimensions. *Journal of Visual Languages & Computing*, 21(5):249–262, 2010.

[8] S.-C. Cheung, W. Chen, Y. Liu, and C. Xu. CUSTODES: Automatic Spreadsheet Cell Clustering and Smell Detection Using Strong and Weak Features. In *Proceedings of the 38th International Conference on Software Engineering (ICSE)*. 2016. To appear.

[9] W. Dou, S.-C. Cheung, and J. Wei. Is Spreadsheet Ambiguity Harmful? Detecting and Repairing Spreadsheet Smells Due to Ambiguous Computation. In *Proceedings of the 36th International Conference on Software Engineering (ICSE)*, pages 848–858. 2014.

[10] P. Durusau and S. Hunting. Spreadsheets - 90+ million End User Programmers With No Comment Tracking or Version Control. In *Proceedings of Balisage: The Markup Conference*. 2015.

[11] M. Fisher and G. Rothermel. The EUSES Spreadsheet Corpus: A Shared Resource for Supporting Experimentation with Spreadsheet Dependability Mechanisms. *ACM SIGSOFT Software Engineering Notes*, 30(4):1–5, 2005.

[12] G. Rothermel, L. Li, C. Dupuis, and M. Burnett. What You See Is What You Test: A Methodology for Testing Form-based Visual Programs. In *Proceedings of the 20th International Conference on Software Engineering (ICSE)*, pages 198–207. 1998.

[13] T. Hall, S. Beecham, D. Bowes, D. Gray, and S. Counsell. A Systematic Literature Review on Fault Prediction Performance in Software Engineering. *IEEE Transactions on Software Engineering (TSE)*, 38(6):1276–1304, 2012.

[14] F. Hermans and E. Murphy-Hill. Enron's Spreadsheets and Related Emails: A Dataset and Analysis. In *Proceedings of the 37th International Conference on Software Engineering (ICSE)*, pages 7–16. 2015.

[15] F. Hermans, M. Pinzger, and A. van Deursen. Supporting Professional Spreadsheet Users by Generating Leveled Dataflow Diagrams. In *Proceedings of the 33th International Conference on Software Engineering (ICSE)*, pages 451–460. 2011.

[16] F. Hermans, M. Pinzger, and A. van Deursen. Detecting and Visualizing Inter-worksheet Smells in Spreadsheets. In *Proceedings of the 34th International Conference on Software Engineering (ICSE)*, pages 441–451. 2012.

[17] F. Hermans, B. Sedee, M. Pinzger, and A. van Deursen. Data Clone Detection and Visualization in Spreadsheets. In *Proceedings of the 35th International Conference on Software Engineering (ICSE)*, pages 292–301. 2013.

[18] B. Jansen and F. Hermans. Code Smells in Spreadsheet Formulas Revisited on an Industrial Dataset. In *31st International Conference on Software Maintenance and Evolution (ICSME)*. 2015.

[19] D. Kim, J. Nam, J. Song, and S. Kim. Automatic Patch Generation Learned from Human-written Patches. In *Proceedings of the 35th International Conference on Software Engineering (ICSE)*, pages 802–811. 2013.

[20] M. Kim, V. Sazawal, D. Notkin, and G. Murphy. An Empirical Study of Code Clone Genealogies. In *Proceedings of the 10th European Software Engineering Conference Held Jointly with 13th ACM SIGSOFT International Symposium on Foundations of Software Engineering (ESEC/FSE)*, pages 187–196. 2005.

[21] S. Kim, T. Zimmermann, E.J. Whitehead Jr., and A. Zeller. Predicting Faults from Cached History. In *Proceedings of the 29th International Conference on Software Engineering (ICSE)*, pages 489–498. 2007.

[22] B. Klimt and Y. Yang. Introducing the Enron Corpus. In *First Conference on Email and Anti-Spam (CEAS) in Cooperation with AAAI and The International Association for Cryptologic Research and The IEEE Technical Committee on Security and Privacy*. 2004.

[23] H.A. Nguyen, T.T. Nguyen, N.H. Pham, J. Al-Kofahi, and T.N. Nguyen. Clone Management for Evolving Software. *IEEE Transactions on Software Engineering (TSE)*, 38(5):1008–1026, 2012.

[24] R. Panko. Facing the Problem of Spreadsheet Errors. *Decision Line*, 37(5):8–10, 2006.

[25] R.R. Panko and S. Aurigemma. Revising the Panko–Halverson taxonomy of spreadsheet errors. *Decision Support Systems*, 49(2):235–244, 2010.

[26] S.G. Powell, K.R. Baker, and B. Lawson. A Critical Review of the Literature on Spreadsheet Errors. *Decision Support Systems*, 46(1):128–138, 2008.

[27] K. Rajalingham, D.R. Chadwick, and B. Knight. Classification of Spreadsheet Errors. *arXiv:0805.4224 [cs]*, 2008.

[28] J. Reichwein, G. Rothermel, and M. Burnett. Slicing Spreadsheets: An Integrated Methodology for Spreadsheet Testing and Debugging. *ACM SIGPLAN Notices*, 35(1):25–38, 1999.

[29] C. Scaffidi, M. Shaw, and B. Myers. Estimating the Numbers of End Users and End User Programmers. In *Proceedings of the IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, pages 207–214. 2005.

[30] S. Thummalapenta, L. Cerulo, L. Aversano, and M. Di Penta. An Empirical Study on the Maintenance of Source Code Clones. *Empirical Software Engineering*, 15(1):1–34, 2010.

[31] Spreadsheet Compare. https://technet.microsoft.com/en-us/library/dn205148.aspx. [Accessed: 7-Feb-2016].

[32] Apache POI - the Java API for Microsoft Documents. http://poi.apache.org/. [Accessed: 7-Feb-2016].