Delft University of Technology
Software Engineering Research Group
Technical Report Series

Detecting Problematic Lookup Functions in Spreadsheets

Felienne Hermans, Efthimia Aivaloglou, Bas Jansen

Report TUD-SERG-2015-011

TUD-SERG-2015-011

Published, produced and distributed by:

Software Engineering Research Group
Department of Software Technology
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology
Mekelweg 4
2628 CD Delft
The Netherlands

ISSN 1872-5392

Software Engineering Research Group Technical Reports:
http://www.se.ewi.tudelft.nl/techreports/

For more information about the Software Engineering Research Group:
http://www.se.ewi.tudelft.nl/

Note: Accepted for publication in the Proceedings of the IEEE Symposium in Visual Languages and Human Centric Computing (VL/HCC 2014)

Detecting Problematic Lookup Functions in Spreadsheets

Felienne Hermans, Efthimia Aivaloglou, Bas Jansen
Delft University of Technology
{f.f.j.hermans, e.aivaloglou, b.jansen}@tudelft.nl

***Abstract*—Spreadsheets are used heavily in many business domains around the world. They are easy to use and as such enable end-user programmers to and build and maintain all sorts of reports and analyses. In addition to using spreadsheets for modeling and calculation, spreadsheets are often also used for creating reports and dashboards: combining data from different sources and creating overviews. For this, *lookup functions* can be used: they search for a value in a range and return a corresponding row or column. Lookup functions are common: according to recent research the VLOOKUP is the fifth most common Excel function. In this paper we investigate the use of lookup functions in more detail. We analyze lookup functions within the newly released Enron spreadsheet corpus. The results show that 1) a minority of 43% of lookup formulas use the default setting where an approximate match may be returned, 2) 77% of approximate matches are used unnecessary and 3) 23% of approximate lookups is problematic: they search over unsorted ranges, while this is specifically advised against in the specification, and might lead to wrong results.**

I. Introduction

Spreadsheets are used in many different business domains, from finance and logistics to planning and operations. They can be considered the most successful end-user programming paradigm in history, with an estimated 23 million professional users in America, amounting to about 30% of the workforce [1]. Their use is diverse; some spreadsheets are used as simple "flat lists", without any calculations in them, while others implement complex algorithms using different formulas. In this paper we investigate the the use of a specific type of function in spreadsheets: *lookup functions*. They exist in a few different variations, all aimed at looking up a value within a range.

In order to better understand the use of lookup functions, we analyze their use in the Enron corpus, a recently released set of more than 16.000 spreadsheets from the bankrupt company Enron [2]. We are especially interested in learning more about the two different ways in which lookup functions can be applied: for exact matching, where only exactly corresponding results can be returned—often used to combine two worksheets—and the approximate match, where approximate results may be returned, used mainly for simple classification.

Our results show that lookup functions present spreadsheet developers with several issues. Firstly, a minority of 43% of lookup formulas use the default setting. More interestingly, in 77% of the cases where approximate matches is employed, it was not necessary. Finally, 23% of approximate lookups search over unsorted ranges, while this is specifically advised against in the specification, as it might lead to wrong results. We identified 11 formulas in the set that resulted in an erroneous value due to this issue.

II. Background

Most spreadsheet systems support lookup functions, meant to search for data, allowing users to combine data from different sources Lookup functions go way back: VisiCalc, widely considered to the world's first program spreadsheet system, already supported LOOKUP as one of its 20 built-in functions, which is quite remarkable as it lacked many other common spreadsheet functions, including IF. The most well-known use of LOOKUP functions is as a means to couple two tables, comparable to a JOIN operation in SQL like languages. This use is shown in Figure 1, for every 'Code' in column F the corresponding 'Major' in column G is returned.

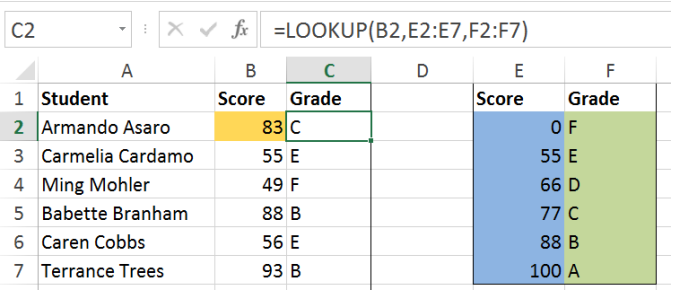
While this is a common use of lookup functions in spreadsheets, there is a second way in which they can be used: for approximate matching. Figure 2 shows this type of matching. The selected formula in C2 here searches for 83 and returns 'C', because 77 which appears on the the same row as 'C', is the highest value below 83.

D2 =LOOKUP(C2,F2:F6,G2:G6)

	A	B	C	D	E	F	G
1	Id	Student	Code	Major		Code	Major
2	13762	Armando Asaro	3	Civil Eng		1	Aero Eng
3	19876	Carmelia Cardamo	5	Mech Eng		2	Elec Eng
4	14833	Ming Mohler	5	Mech Eng		3	Civil Eng
5	17257	Babette Branham	3	Civil Eng		4	Comp Sci
6	16075	Caren Cobbs	1	Aero Eng		5	Mech Eng
7	17437	Terrance Trees	4	Comp Sci			

Fig. 1. A LOOKUP function, used in this example to couple two tables

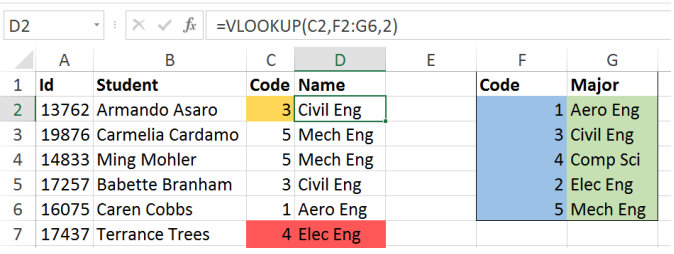
In addition to the simple LOOKUP function, most modern spreadsheet systems support other, more powerful lookup functions as well. The most common lookup function is VLOOKUP, which always searches for a value in a vertical search range, i.e. one column. VLOOKUP needs three arguments: a search value: either a value or a reference to a cell with a value, a range in which to search, and a column *number* indicating which column to return from. The most notable difference between LOOKUP and VLOOKUP is that the third parameter is an integer rather than a range. A VLOOKUP equal to the formula in Figure 2 is VLOOKUP(B2,E2:E7,2).



	A	B	C	D	E	F
1	Student	Score	Grade		Score	Grade
2	Armando Asaro	83	C		0	F
3	Carmelia Cardamo	55	E		55	E
4	Ming Mohler	49	F		66	D
5	Babette Branham	88	B		77	C
6	Caren Cobbs	56	E		88	B
7	Terrance Trees	93	B		100	A

Fig. 2. A LOOKUP function, used in this example to find an approximate match

In addition to the three required parameters, VLOOKUP can be called with an optional fourth parameter of type Boolean that indicates happens when the search value is not found in the search range. When this parameter is set to FALSE only exact matches will be returned; an #N/A error is returned when the search value is not found. If we would change the formula in Figure 2 to VLOOKUP(B2,E2:E7,2, FALSE) it would result in #N/A in the second row, as 83 does not occur in the search range E2:F7. TRUE is the default value for the fourth parameter, i.e. using TRUE as parameter results in the same behavior as omitting it. In addition to VLOOKUP, most spreadsheet systems also support HLOOKUP, a function identical to VLOOKUP apart from searching in and returning from a horizontal range.



	A	B	C	D	E	F	G
1	Id	Student	Code	Name		Code	Major
2	13762	Armando Asaro	3	Civil Eng		1	Aero Eng
3	19876	Carmelia Cardamo	5	Mech Eng		3	Civil Eng
4	14833	Ming Mohler	5	Mech Eng		4	Comp Sci
5	17257	Babette Branham	3	Civil Eng		2	Elec Eng
6	16075	Caren Cobbs	1	Aero Eng		5	Mech Eng
7	17437	Terrance Trees	4	Elec Eng			

Fig. 3. A VLOOKUP function approximately searching in an unsorted range, resulting in erroneous values

For the remainder of the paper, it is important to know that approximate matching only works when the search range is sorted, as it relies on binary search when searching. When the range is not sorted, approximate lookups, both for HLOOKUP and VLOOKUP, will 'fail silently'. As an example consider the spreadsheet depicted in Figure 3, where a user has sorted the cells in F2:G6 on the names of the majors, rather than on the majors's code, hence the search range in F2:F6 is not sorted. Now, the VLOOKUP function in column *D* returns a wrong major for code 4, as highlighted in red in cell D7. When this occurs, no error message or warning is given. This is a known issue with spreadsheet lookup functions, officially documented by Microsoft[1], but also occuring in other spreadsheet systems, including LibreOffice and Google Spreadsheets.

[1]https://support.microsoft.com/en-us/kb/181201

TABLE I
THE NUMBER OF UNIQUE FORMULAS IN THE ENRON CORPUS FOR THE THREE DIFFERENT OPTIONS THAT LOOKUP FUNCTIONS HAVE: WITH 3 PARAMETERS, WITH 4 PARAMETERS USING TRUE (TOGETHER REPRESENTING THE NUMBER OF APPROXIMATE MATCHES) AND WITH 4 PARAMETERS USING FALSE.

Fourth parameter		T	Total Approx.	F	Total
# parameters	3	4		4	
VLOOKUP	5,187	11	5,198	7,010	12,208
HLOOKUP	956	20	976	1,047	2,023
Total	6,143	31	6174	8,057	14,231

III. SMELLS IN LOOKUP FUNCTIONS

While researching spreadsheets for the past years, we have often seen that lookup functions can be error-prone. Recently, we released the Enron corpus [2], enabling us to answer research questions focusing on the use of lookup functions on a large set of industrial spreadsheets. In our research we focus on the two different settings for searching: approximate and exact and on situations in which issues with lookup functions arise. We thus focus on the following research questions:

1) How often are the two different settings, approximate versus exact, used?
2) How often do spreadsheet users use approximate match while in reality an exact match is being performed?
3) How often do users spreadsheet users use approximate match within an unsorted search range?

A. How often are the two different settings (approximate versus exact) used?

Remember that the lookup functions VLOOKUP and HLOOKUP can be used with three parameters, but also have an optional fourth parameter, indicating whether an approximate match or an error should be returned, when the search value is not found. To understand the issues around lookups, we first investigate how common the different usages are: using 3 parameters, using 4 with TRUE and using 4 with FALSE.

Table I shows the result of our analysis. It shows the number of unique formulas for all three different options that lookup functions have: with 3 parameters, with 4 parameters using TRUE (together representing the number of approximate matches) and with 4 parameters using FALSE.

The first thing to note is that VLOOKUP is much more commonly used than HLOOKUP, indicating that most spreadsheets have a layout where rows represent items. Secondly, we see that, while not being the default, exact matches are more common than approximate ones. In VLOOKUP the numbers are 5,198 approximate versus 7,010 exact and for HLKOOP those numbers are 976 versus 1,047. Hence of the total 14,231 lookup functions, 43% is approximate and 57% is exact. Apparently, lookup functions are used in a different way than was envisioned when they were introduced. We have inquired with the Excel team when VLOOKUP was introduced

and they told us that was in Excel 5 in 1997. Before that, Excel only supported LOOKUP.

> Approximate match, while being the default match setting, is not the most common setting in lookup functions, as only 43% of lookup formulas uses this setting.

B. How often do spreadsheet users use approximate match while in reality an exact match is being performed?

It is interesting that in the Enron corpus we observe that the exact setting is used more often than the approximate match, while it is not the default. However, having worked with spreadsheet users extensively over the past years, we were somewhat surprised that the exact setting was not used *even more frequently*. This lead us to inspect the cases of approximate matching in more detail.

We searched for cases in which the approximate match was used, but where it was not 'needed', i.e. there were no values searched for in between the values in the lookup range. An example is shown in Figure 4, where the values in the search range are 0..16, and these are exactly the values that are being sought for, in the formulas in B24:R24 (searching for the values in B5:R5 which are exactly 0..16).

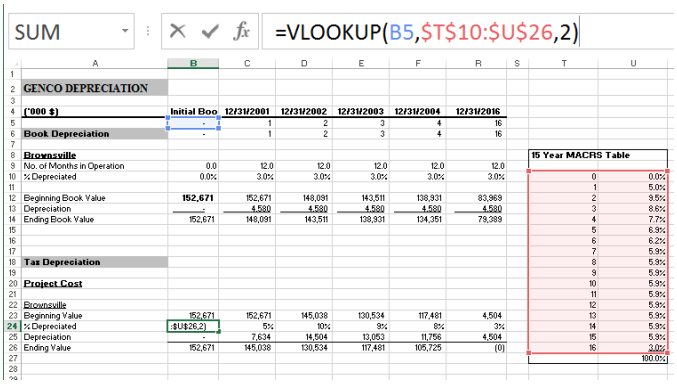


Fig. 4. A VLOOKUP function using the approximate setting, but only searching for values occurring in the search range.

Of course, this could be a deliberate choice by the spreadsheet's designer. Using the approximate match setting is more efficient, because binary search is used instead of linear search. However, especially for small search ranges, this will not make an observable difference. Furthermore is doubtful whether spreadsheets users are aware of such intricacies in the spreadsheet's execution engine. A more likely explanation is that the approximate setting was used by mistake. We found that in a staggering 4,792 (77.6%) of all approximate matches, the approximate setting was not needed. We assert this use as problematic, because, when a value to search for is added or changed, the approximate match will result in an approximate value instead of an error.

> 77% of approximate match formulas use the approximate setting while it is not needed.

C. How often do users spreadsheet users use approximate match within an unsorted search range?

As described above, when a lookup function is set to approximate, either by using TRUE as fourth parameter or by omitting the fourth parameter, it will only work correctly if the data in which the lookup searches is sorted. Excel, but also other spreadsheet systems, including Google Docs and LibreOffice, do not warn the user when this effect occurs.

Hence, we are interested in cases where an approximate match is used in combination with a search in an unsorted range, as this might result in erroneous values. When analyzing the 16,270 unique lookup formulas in the Enron Corpus, we found that 1,437 approximate matches searches over an unsorted search range, which is about 23% of the approximate matches in the set. In the following subsections we will describe some of the patterns we observed.

1) Including Headers: By far the most common case in which search ranges were unsorted is the case where a header was included the search range, as in Figure 5.

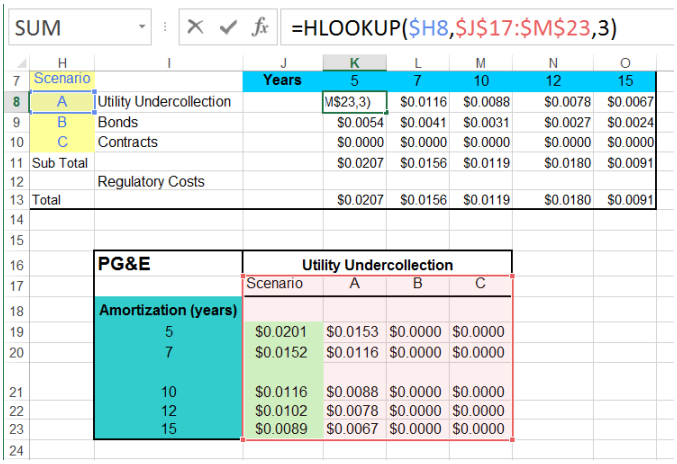


Fig. 5. A VLOOKUP function including the header row.

As you can see in this figure, the lookup function searches in the first row—J17:M17—and while doing so also includes *'Scenario'*, which is a header rather than being a value sought. The inclusion of *'Scenario'* into the search range means that the search range is not sorted, which can result in strange behavior. For example, if a user would expand the search range with one additional scenario, scenario "A" can not be located anymore. We imagine this is an error that is hard to resolve for Excel users unfamiliar with the exact inner workings of spreadsheet systems.

An interesting variation occured in a file where one of the values under lookup is "0.5 or lower". Presumably, the spreadsheet users here thought that values under 0.5 will be matched with 0.04762, but, in reality, all values under and including will result in #N/A.

2) Actual mistakes: Most of the cases we found for unsorted lookups were mere 'smells': accidents waiting to happen, but not happened yet. Some of the cases however, already took a turn for the worst. A remarkably deceiving example is

shown in Figure 6. Because the postfix " year" is added to values in the lookup range, Excel views all values as string. Hence, the column, while appearing sorted, is not.

I48 =VLOOKUP(H48,C48:C59,1)

	C	D	E	F	G	H	I
47		Spread over US-T (Bps)	27-Jul Yield	Current Yield			
48	1 year	35	3.84	3.68		1 year	1 year
49	2 year	54	4.34	4.26		2 year	2 year
50	3 Year	75	4.81	4.74		3 Year	3 Year
51	4 Year	80	5.14	5.07		4 Year	4 Year
52	5 year	76	5.37	5.30		5 year	5 year
53	6 year	67	5.53	5.47		6 year	6 year
54	7 year	74	5.65	5.60		7 year	7 year
55	8 year	78	5.75	5.69		8 year	8 year
56	9 year	77	5.83	5.76		9 year	9 year
57	10 year	75	5.89	5.83		10 year	1 year
58	15 year	92	6.13	6.06		15 year	1 year
59	30 year	68	6.27	6.21		30 year	3 Year
60	Source: Bloomberg						

Fig. 6. Lookup is being performed over range C48:C59, which contains values that appear sorted, but are not. When performing a lookup over this range, as illustrated by the right table, wrong values are returned (marked in red)

This leads to interesting errors, as illustrated by range H48:I59 in Figure 6, added by the authors of this paper for illustrative purposes; it does not occur in the original file. When looking up years 10, 15 and 30, erroneous results are returned, in this case causing the yield to be underestimated by about 30%, without any warning being given to the user.

While the number of mistakes was small, 11 formulas in 8 files, they all resulted in incorrect results in the spreadsheets.

> Almost a quarter (23%) of approximate matches searches over an unsorted search range, which is incorrect according to the official specification. In 11 of the cases, this even leads to wrong results.

IV. Related Work

The issues we address in this paper are examples of 'code smells' applied to spreadsheets. Code smells were initially started by Fowler [3] that gives an overview of code smells and corresponding refactorings. We ourselves have worked on spreadsheet smells in previous work, by detecting smells between worksheets like high coupling [4], but also smells at the formula level, like conditional complexity [5]. Following the work on spreadsheet smells, we have worked on refactorings for spreadsheets also, by defining refactorings accompanying our smells [6], which followed our earlier work, in which we worked on the visualization of spreadsheets by means of class diagrams [7] and dataflow diagrams [4]. Based on the work on the refactoring approach, we defined a generic method to describe and execute refactorings in spreadsheets [8], which combined our smells with those defined by Badame and Dig [9].

Related also is work on mining of corpora of source code to locate deviations from the specification. For example the work of Wasylkowski *et al.* that examines code examples to automatically infer legal sequences of method calls. The resulting patterns can then be used to detect anomalies [10]. The work of Wasylkowski followed an earlier approach by Li and Ahou [11] that also extracted programming rules and used them to automatically detect violations to the extracted rules.

V. Conclusions

This paper focuses on the use of and the problems associated with lookup functions in spreadsheets: functions aimed at combining information over different worksheets or even files. While these functions are popular—VLOOKUP is the fifth most common function in the Enron set—we have found they are error prone too. In this paper, we have addressed the following research questions:

- How often are the two different settings (approximate versus exact) used?
- How often do spreadsheet users specify the wrong type of match?
- How often do users spreadsheet users use approximate match over an unsorted search range?

Our results show that lookup functions present spreadsheet developers with several issues. Firstly, a minority of 43% of lookup formulas use the default setting, which is approximate matching. This shows the default might not have been chosen in the best way. More interestingly, in 77% of the cases where approximate match is used, it was not necessary, since no approximate values were being sought. Finally, 23% of approximate lookups is *smelly*: they search over unsorted ranges, while this is specifically advised against in the specification, as it might lead to wrong results.

The current research gives rise to several avenues for future work. First of all, we would like to implement detection of lookup smells into our existing code base for spreadsheet analysis [12] and subsequently perform a user study to determine if lookup smells detection is usable and scalable for real life spreadsheets and their users. Furthermore, it would be useful to implement refactorings for smelly LOOKUP functions, and subsequently create more involved refactorings, such as changing a lookup function into a combination of the INDEX and MATCH functions, which are more efficient, or introducing a lookup when a direct link is used.

REFERENCES

[1] C. Scaffidi, M. Shaw, and B. A. Myers, "Estimating the numbers of end users and end user programmers," in *Proc. of VL/HCC '05*, 2005, pp. 207–214.

[2] F. Hermans and E. Murphy-Hill, "Enron's spreadsheets and related emails: A dataset and analysis," in *37th International Conference on Software Engineering, ICSE '15*, to appear.

[3] M. Fowler, *Refactoring: improving the design of existing code*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1999.

[4] F. Hermans, M. Pinzger, and A. van Deursen, "Detecting and visualizing inter-worksheet smells in spreadsheets," in *Proc. of ICSE '12*, 2012, pp. 441–451.

[5] ——, "Detecting code smells in spreadsheet formulas," in *Proc. of ICSM '12*, 2012, pp. 409–418.

[6] ——, "Detecting and refactoring code smells in spreadsheet formulas," *Empirical Software Engineering*, pp. 1–27, 2014.

[7] ——, "Automatically extracting class diagrams from spreadsheets," in *Proc. of ECOOP '10*, 2010, pp. 52–75.

[8] F. Hermans and D. Dig, "Bumblebee: a refactoring environment for spreadsheet formulas," *Proceedings of the 22nd ACM SIGSOFT Symposium on the Foundations of Software Engineering (FSE '14)*, pp. 747–750, 2014.

[9] S. Badame and D. Dig, "Refactoring meets spreadsheet formulas," in *Proc. of ICSM '12*, 2012, pp. 399–409.

[10] A. Wasylkowski, A. Zeller, and C. Lindig, "Detecting object usage anomalies," in *Proceedings of the the 6th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on The Foundations of Software Engineering*, ser. ESEC-FSE '07. New York, NY, USA: ACM, 2007, pp. 35–44. [Online]. Available: http://doi.acm.org/10.1145/1287624.1287632

[11] Z. Li and Y. Zhou, "Pr-miner: Automatically extracting implicit programming rules and detecting violations in large software code," in *Proceedings of the 10th European Software Engineering Conference Held Jointly with 13th ACM SIGSOFT International Symposium on Foundations of Software Engineering (ESEC-FSE '13)*, ser. ESEC/FSE-13. New York, NY, USA: ACM, 2005, pp. 306–315. [Online]. Available: http://doi.acm.org/10.1145/1081706.1081755

[12] F. Hermans, "Analyzing and visualizing spreadsheets," Ph.D. dissertation, Delft University of Technology, 1 2013. [Online]. Available: http://www.felienne.com/?p=2534

TUD-SERG-2015-011
ISSN 1872-5392