2015 12th Working Conference on Mining Software Repositories

FUSE: A Reproducible, Extendable, Internet-scale Corpus of Spreadsheets

Titus Barik*†, Kevin Lubick†, Justin Smith†, John Slankas†, Emerson Murphy-Hill†
*ABB Corporate Research, Raleigh, North Carolina, USA
†North Carolina State University, Raleigh, North Carolina, USA
titus.barik@us.abb.com, {kjlubick, jssmit11, jbslanka}@ncsu.edu, emerson@csc.ncsu.edu

***Abstract*—Spreadsheets are perhaps the most ubiquitous form of end-user programming software. This paper describes a corpus, called FUSE, containing 2,127,284 URLs that return spreadsheets (and their HTTP server responses), and 249,376 unique spreadsheets, contained within a public web archive of over 26.83 billion pages. Obtained using nearly 60,000 hours of computation, the resulting corpus exhibits several useful properties over prior spreadsheet corpora, including reproducibility and extendability. Our corpus is unencumbered by any license agreements, available to all, and intended for wide usage by end-user software engineering researchers. In this paper, we detail the data and the spreadsheet extraction process, describe the data schema, and discuss the trade-offs of FUSE with other corpora.**

I. INTRODUCTION

End-user programmers today constitute a broad class of users, including teachers, accountants, administrators, managers, research scientists, and even children [1]. Although these users are typically not professional software developers, their roles routinely involve computational tasks that, in many ways, are similar to those of developers — not just in activity, but also in their underlying cognitive demands on users [2].

Perhaps the most ubiquitous form [3] of end-user programming software are *spreadsheets*, a table-oriented visual interface that serves as the underlying model for the users' applications [4]. *Cells* within these tables are augmented with computation, such as expressions, functions and macros [4]. This interplay between presentation and computation within the spreadsheet environment has garnered significant interest from the software engineering research community [5]. Researchers have adopted techniques and approaches to studying errors [6], code smells [7], and refactoring in spreadsheets [8], similar to traditional programming environments.

To better understand end-user activities and design tools to assist end-users, researchers have responded by curating spreadsheet corpora to support spreadsheet studies [9], [10], [11]. This paper presents one such spreadsheet corpus, called FUSE, extracted from the over 26.83 billion web pages in the Common Crawl[1] index. We believe that FUSE offers several useful traits not found in prior corpora; for example, FUSE is obtained in such a way that researchers can independently reproduce an identical corpus from source materials.

[1]The Common Crawl non-profit organization provides this index to companies and individuals at no cost for the purpose of research and analysis. For more information, see http://www.commoncrawl.org.

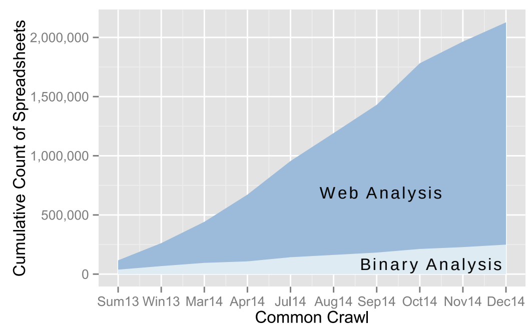

Fig. 1. Cumulative count of spreadsheets obtained with each additional crawl. Web Analysis contains all URLs and associated HTTP server responses, while Binary Analysis contains the actual spreadsheets for a subset of Web Analysis, archived within Common Crawl.

The contributions of this paper are two related datasets, which together constitute the FUSE spreadsheet corpus[2]:

- A *Web Analysis* dataset of 2,127,284 URLs that return spreadsheet content, along with the full HTTP web server response, formatted as JSON records. This dataset is obtained by filtering through 26.83 billion HTTP responses within the Common Crawl archive.
- A *Binary Analysis* dataset of 249,376 spreadsheets, extracted from the 1.9 PB of raw data within the Common Crawl archive. For each spreadsheet, we provide JSON metadata containing our analysis, which includes NLP token extraction and spreadsheet metrics.

II. DESCRIPTION OF DATA

The Common Crawl index contains not only the HTTP responses of web pages, but also the raw content of each of these resources, including binaries. It is from these monthly web crawls that we extract and make available spreadsheets and corresponding metadata, augmented with our analysis, and tailored to researchers.

The result, FUSE, is characterized through two, hierarchical datasets (Figure 1): a Web Analysis dataset, and a Binary Analysis dataset.

Web Analysis: This dataset contains 2,127,284 spreadsheet-related URLs and HTTP responses. 292,043 of these responses

[2]The corpus metadata, spreadsheets, tools, and other documentation can be obtained at http://go.ncsu.edu/fuse.

978-0-7695-5594-2/15 $31.00 © 2015 IEEE
DOI 10.1109/MSR.2015.70

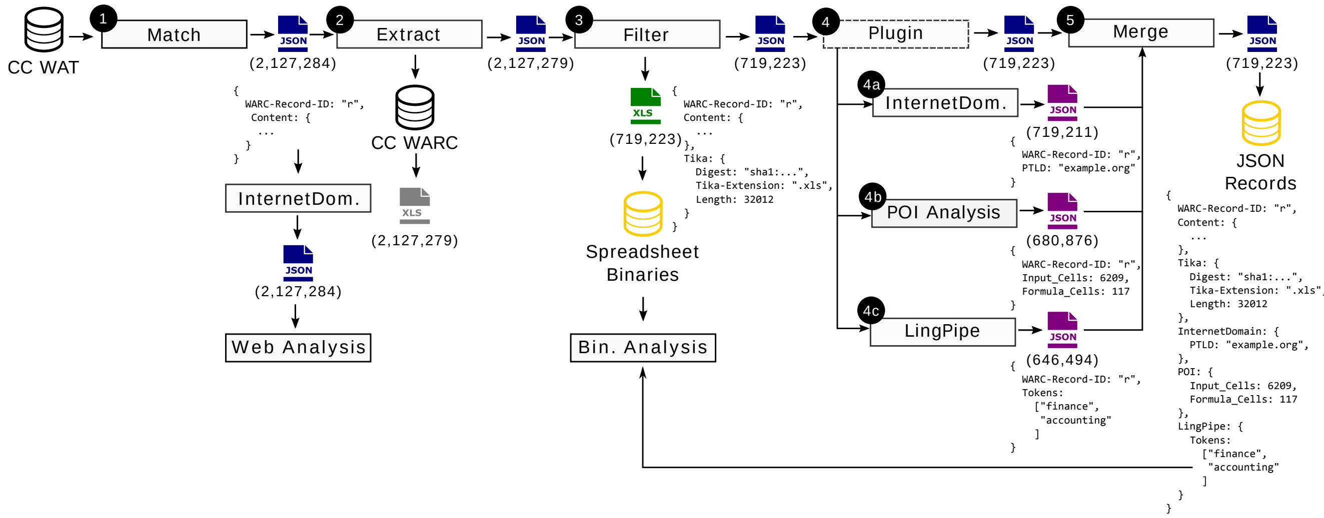

Fig. 2. The MapReduce pipeline for extracting spreadsheets and associated spreadsheet analysis metadata from Common Crawl.

point to a unique URL, and the top domain is `.org` (29.5%), followed by `.gov` (27.7%). The analysis contains 6,316 distinct domain names. Unfortunately, relying solely on Web Analysis for spreadsheets will not result in a reproducible corpus, as the Internet is always in flux.

Binary Analysis: To address the limitations of Web Analysis, the Binary Analysis dataset contains 249,376 unique spreadsheets, extracted directly from the raw data contained within Common Crawl archives, rather than the Internet. Since each monthly Common Crawl archive is a permanent snapshot in time, Binary Analysis is always reproducible.

Analyzing these spreadsheets, we discovered that `IF` is the most frequently used function, found in 17.8% of all formula cells, giving evidence that spreadsheets require non-trivial computation. We also discovered that `=SUM(R[-3]C:R[-1]C)` is the most common formula, in which a cell is the sum of the three cells to its left, and that it appears in 1,322 spreadsheets. In contrast with domain specific corpora, such as Enron [10], our general spreadsheet corpus has fewer formulas. Interestingly, only 7.00% of our spreadsheets contain any formula, as opposed to 59.4% of Enron spreadsheets, which is consistent with anecdotal findings of the Excel team.[3]

This analysis hierarchy has several properties desirable to researchers, the first of which is reproducibility. In Web Analysis, an independent researcher should always obtain the same set of spreadsheet-related URLs, provided they use the same spreadsheet detection heuristic. Because the spreadsheets from Binary Analysis are obtained from content embedded in the Common Crawl corpus, they too are reproducible resources. A second property of our corpus is that it is open to extension, without sacrificing reproducibility. When Common Crawl releases a new dataset, these crawls can be incrementally incorporated into FUSE. A third useful property of our corpus is related not to the data itself, but to its broader ecosystem: FUSE is unencumbered by any licensing requirements, available to all, and includes a scalable, open source toolchain.

[3]Joel Spolsky writes, "Everybody thought of Excel as a financial modeling application, [but] we visited dozens of Excel customers, and did not see anyone using Excel to actually perform what you would call 'calculations.' Almost all of them were using Excel because it was a convenient way to create a table. [12]"

III. Methodology

The Common Crawl is available as a public dataset on Amazon.[4] The crawl data is hosted on Simple Storage Service (S3) as a set of WARC (Web ARChive) files, which store the raw crawl data, and corresponding WAT files, which store the web crawl metadata for a given WARC file. Essentially, each WAT file contains JSON-formatted records that act as an index into the WARC raw data. That is, each record contains a globally unique identifier, called the `WARC-Record-ID`, and a reference to a WARC filename, offset, and length. S3 supports downloading segments of files in this way.

We examined all crawl data stored in WARC format. At the time of analysis, this covered the period beginning with Summer 2013 through December 2014. This period consists of 26.83 billion web pages, compressed to 423.8 TB (1.9 PB uncompressed). To support parallelization, this data is split into 481,427 segments, such that different machines can independently process a segment. Extracting such a corpus from a single desktop machine is computationally intractable, and thus we extracted the spreadsheets using the Amazon Elastic MapReduce service.

A. Hadoop MapReduce Pipeline

Figure 2 illustrates our MapReduce framework, which consists of five stages that comprise a pipeline. For each stage in the framework, we compute the cost in terms of normalized instance hours. The total hours correspond to the approximate amount of time that it would take for a 1 vCPU, 1.7

[4]http://aws.amazon.com/datasets/41740

GiB machine to complete the task — in other words, roughly comparable to a single end-user desktop machine. Although researchers do not need to use our pipeline to reproduce our results, our framework already contains the necessary MapReduce scaffolding, such as task scheduling code, as well as Java libraries, to support distributed analysis.

1) Match: This stage required that we traverse every JSON-formatted URL and HTTP response in the 481,427 WAT segments and match spreadsheet-related records. First, we checked if the HTTP response payload `Content-Type` field corresponded to one of seven spreadsheet MIME types as supported by Microsoft Excel.[5] However, some records contained a generic binary `Content-Type` of `application/octet-stream`, in which case `Content-Disposition` was checked via a file pattern matching "`.xls*`". If either of these conditions were true, we saved the record using the `WARC-Record-ID` as the key. This key identified the file throughout the pipeline. The match stage is a heuristic process because we cannot know for sure that a record is actually a spreadsheet until we inspect the extracted file. After filtering through some 26.83 billion records, we identified 2,127,284 candidate spreadsheets. This stage, the most computationally expensive in the pipeline, required about 55,000 normalized instance hours to process.

2) Extract: The extract stage loaded the 2,127,284 candidate spreadsheet records. Using the `Filename`, `Offset`, and `Deflate-Length` fields of the record, the corresponding WAT record was extracted into memory. The WARC record was then stripped of its header information (e.g., the HTTP response), and the remaining content was saved to S3, again using the `WARC-Record-ID` from the WAT file as the key. Theoretically, this process should yield the same number of records as the match stage; however, five records had corrupted `gzip` entries, yielding 2,127,279 candidate spreadsheets. This stage required about 1,000 normalized instance hours to complete.

3) Filter: The filter stage checked the extracted file and tagged those that were actually spreadsheets. We used Apache Tika[6], a content analysis library, to detect the `Content-Type` of the file. If this result was one of the spreadsheet MIME types, the record was retained. During this stage, we also computed the length (in bytes) of the spreadsheet, identified the most appropriate file extension (e.g., "`.xlsx`"), and generated a SHA-1 digest of the spreadsheet content. At this stage, 719,223 spreadsheets were retained in the pipeline, although many of these were duplicates. This stage required about 420 normalized instance hours to complete.

4) Plugins: The fourth stage of the pipeline is actually a meta-stage, in which researchers can augment the framework using plugins for their own analysis. For our corpus, we augmented the JSON record with three plugins: `InternetDomain`, which uses the Google Guava[7] library to extract domain-related information from the `WARC-Target-URI`; Apache `POI`[8], which obtains metrics on the content of the spreadsheets, such as the use of functions; and `LingPipe`[9], which extracts language-related information from the spreadsheet. These JSON records were all saved to S3 by their `WARC-Record-ID`. This stage required about 400 normalized instance hours per plugin. Researchers wishing to build on our approach will be able to insert their own plugins at this stage, without having to recompute the first three stages, saving research time and effort. For convenience, we also retroactively ran the `InternetDomain` plugin on the JSON output from the match stage to generate the Web Analysis dataset.

5) Merge: The merge stage simply took the resulting JSON files from all previous stages and combined them with the original WAT record to facilitate downstream analysis. This stage required about 130 normalized instance hours for each plugin. The output of merge, combined with the spreadsheets from the filter stage, comprise the Binary Analysis dataset.

B. Local Operations

We used the SHA-1 digests from the filter stage and performed a local (non-MapReduce), deterministic, de-duplication operation on the Binary Analysis. The result of this operation was 249,376 unique spreadsheets. Locally, we also concatenated all JSON records to a format readable by MongoDB[10].

IV. Data Schema

We provide the entire data schema as JSON-formatted records. An individual record contains the elements of the original Common Crawl WARC record for that spreadsheet, merged with our analysis results as the record propagated through the pipeline (Figure 2).

The most relevant elements from the Common Crawl WARC record are `WARC-Target-URI`, that is, the URL from which the spreadsheet can be downloaded, and `Container`, which indicates the Common Crawl file and offset used to extract the spreadsheet from the raw crawl data. The `WARC-Date` element may also be of interest, since it contains the time and date of the access. Using the `Content-Disposition` element, one can often extract the original spreadsheet file name.

The `Tika` element contains four fields related to the file metadata, which include the MIME type, a best-guess file extension, a SHA-1 signature, and the length in bytes of the spreadsheet.

The `InternetDomain` element is useful for analysis relating to the origin of a spreadsheet. It uses the `WARC-Target-URI` and extracts the host (e.g., `www.example.org`), the top private domain (e.g., `example.org`), and a public suffix[11] (e.g., `org`).

Next, we provide an Alias-i `LingPipe` element, which extracts the token stream (keywords) from spreadsheets, lowercases these tokens, removes English stop words (such as 'a' or 'the'), and filters out non-words (such as numbers).

Finally, to get a high-level overview of the content of the spreadsheets, we used Apache `POI` to gather spreadsheet

[5]https://technet.microsoft.com/en-us/library/ee309278.aspx
[6]https://tika.apache.org/
[7]https://github.com/google/guava
[8]http://poi.apache.org/
[9]http://alias-i.com/lingpipe/
[10]http://www.mongodb.org/
[11]https://publicsuffix.org/

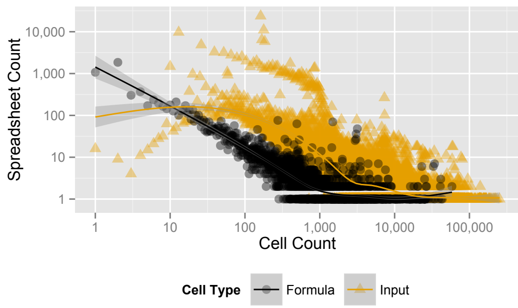

Fig. 3. A log-log distribution of spreadsheets in FUSE by input and formula cell count. 28,616 spreadsheets were unreadable by our analysis tools. For presentation clarity, the graph omits 360 spreadsheets with zero input cells and 205,978 spreadsheets with zero formula cells.

metrics. There are over 450 such metrics, which include the number of times a given Excel function (such as `SUM` or `VLOOKUP`) is used, the total number of input cells (i.e., cells that are not formulas), the number of numeric input cells, and the most common formula used. Figure 3 shows the distributions of formula cells and input cells across all of FUSE. Formula cells and input cells have markedly different distributions.

V. TRADE-OFFS

In this section, we articulate the trade-offs of FUSE in the context of other corpora that provide spreadsheets. The EUSES corpus of 4,498 unique spreadsheets, last updated in 2005, is obtained predominately through parsing the top-ranked Google search results for simple keywords, such as "finance" [9]. In contrast, FUSE has no explicit classification for each spreadsheet, though it may be possible to infer a classification using the `LingPipe` tokens. However, unlike FUSE, EUSES is not reproducible. First, it provides no URL information to obtain the origin for each spreadsheet. Second, the methodology is fundamentally non-deterministic, because Google search results are non-deterministic.

The Enron corpus contains 15,770 spreadsheets extracted from e-mails obtained as legal evidence [10]. Unlike FUSE, Enron is a domain-specific corpus and, consequently, each spreadsheet contains significantly more financial formulas than a general corpus such as ours. In the same vein, FUSE can only offer spreadsheets that are intentionally (or inadvertently) made publicly accessible, and as a result, may contain fewer errors than spreadsheets not for public dissemination. On the other hand, FUSE results suggest that formula-heavy accounting spreadsheets are not representative of general spreadsheet users. Finally, the Enron corpus is forever fixed. In contrast, FUSE can accumulate new URLs and spreadsheets as new Common Crawl datasets are made available.

One limitation is that Common Crawl restricts its storage of binary files to 1 MB. As a result, large spreadsheets are not available in FUSE. On the other hand, spreadsheets larger than 1 MB make up only about 1.0% of EUSES and 2.3% of Enron. However, if one is willing to give up reproducibility, they may use the 292,043 distinct `WARC-Target-URI`s from the Web Analysis and download them using a similar technique as WEB [11]. One advantage of FUSE over WEB is that FUSE contains not only the URL, but also the HTTP response, which includes the crawl access date and `Content-Type` header.

Yet another limitation is that the methodology for Common Crawl is primarily geared towards text-based HTML pages, not binary files. Consequently, any spreadsheets within Common Crawl are only incidental, and not by design. Finally, for various reasons, not all plugins can analyze all spreadsheets, even when they open in Microsoft Excel. For example, our analysis tools do not support pre-1995 (`BIFF5` format) spreadsheets, although we still include them in the corpus.

VI. CONCLUSION

This paper contributes a spreadsheet corpus, FUSE, derived from the Common Crawl. FUSE offers properties not available in existing corpora, including reproducibility and extensibility. Mining software repositories is an inherently cyclic activity: mining data informs insights that require further mining. Our binaries and metadata bootstrap this process, but it is only through custom plugins developed by other researchers that the full potential of FUSE can be realized.

ACKNOWLEDGMENT

This material is based upon work supported in whole or in part with funding from the Laboratory for Analytic Sciences (LAS). Any opinions, findings, conclusions, or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the LAS and/or any agency or entity of the United States Government.

REFERENCES

[1] A. J. Ko, B. Myers, M. B. Rosson, G. Rothermel, M. Shaw, S. Wiedenbeck, R. Abraham, L. Beckwith, A. Blackwell, M. Burnett, M. Erwig, C. Scaffidi, J. Lawrance, and H. Lieberman, "The state of the art in end-user software engineering," *ACM Computing Surveys*, vol. 43, no. 3, pp. 1–44, Apr. 2011.

[2] A. Blackwell, "First steps in programming: A rationale for attention investment models," in *IEEE 2002 Symposia on Human Centric Computing Languages and Environments*, 2002, pp. 2–10.

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

[4] B. A. Nardi and J. R. Miller, "The spreadsheet interface: A basis for end user programming," in *Human-Computer Interaction: INTERACT '90*, 1990, pp. 977–983.

[5] M. Burnett, "What is end-user software engineering and why does it matter?" in *End-User Development SE - 2*, ser. Lecture Notes in Computer Science, 2009, vol. 5435, pp. 15–28.

[6] S. G. Powell, K. R. Baker, and B. Lawson, "A critical review of the literature on spreadsheet errors," *Decis. Support Syst.*, vol. 46, no. 1, pp. 128–138, Dec. 2008.

[7] M. Pinzger, F. Hermans, and A. van Deursen, "Detecting code smells in spreadsheet formulas," in *ICSM '12*, 2012, pp. 409–418.

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

[9] M. Fisher and G. Rothermel, "The EUSES spreadsheet corpus," in *ACM SIGSOFT Software Engineering Notes*, vol. 30, no. 4, Jul. 2005, pp. 1–5.

[10] F. Hermans and E. Murphy-Hill, "Enron's spreadsheets and related emails: A dataset and analysis," in *ICSE SEIP '15*, 2015, to appear.

[11] Z. Chen and M. Cafarella, "Automatic web spreadsheet data extraction," in *Proceedings of the 3rd International Workshop on Semantic Search Over the Web*, Aug. 2013, pp. 1–8.

[12] J. Spolsky. (2012) How Trello is different. [Online]. Available: http://www.joelonsoftware.com/items/2012/01/06.html