A Tool for Visualizing Patterns of Spreadsheet Function Combinations

Justin A. Middleton
North Carolina State University
Raleigh, North Carolina

***Abstract*—Spreadsheet environments often come equipped with an abundance of functions and operations to manipulate data, but it can be difficult to understand how programmers actually use these in practice. Furthermore, users can combine several functions into a complex formula, complicating matters for both researchers and practitioners who want to study formulae to improve spreadsheet practices. Therefore, we developed a tool that visualizes patterns of function combination in spreadsheets as an interactive tree of possibilities. Using spreadsheets from the public spreadsheet corpora, we then apply to the tool to real datasets and demonstrate its ability to capture the most common and most anomalous patterns of function combination and their contexts in actual workbooks.**

I. User Study

Midway through production, we conducted a brief, four-participant user study in order to gauge how well we were capturing the initial design goals. The study was, in general, a flexible and exploratory affair: rather than giving the users any specific tasks or pointed questions, we asked instead that they choose for themselves which trees to explore and report anything that they deemed interesting. Specifically, given trees rooted in the Enron dataset and at least 20 to thirty minutes, the participants situated themselves within the persona of a consultant asked to evaluate a company's spreadsheet practice, which could be arranged around questions like which functions on which the employees most depended, which sheets presented the most anomalous or dangerous designs, and so on. Interpretation of "spreadsheet practice" can certainly vary from participant to participant, but since we wanted a range of views and backgrounds despite the persona, we readily accepted this interpretive wiggle room.

We clarified, furthermore, that these notes could be directed either at the quality of the data conveyed by the tool or at the tool's conveyance itself. Since the tool's philosophy was based in exploration, we decided that it would not have been appropriate to restrict what people were to look for, letting them find interesting stories of formula construction for themselves. Additionally, either kind of comment – on data or on tool – would direct us back to the central evaluation; a user's silence, or lack of any interesting findings, would tacitly indicate that perhaps the tool does not convey information well enough.

Still, several limitations cropped up during the study that complicate the evaluation. For one thing, though the source of the data was not explicitly stated at the outset of the session, several users studied the file names (unchanged as they were) to deduce its roots in finance, if not Enron itself. Because of these notorious connotations, then, some were inclined from this moment of discovery to orient their exploration around finance-related functions, potentially limiting their findings or perceived users for the tool – though some also said that, without this essential context, they could not have properly explored a dataset in the first place. Comments like these point out the trade-offs of unguided exploration, too: though it might not restrict them to a single purpose, it might, at the same time, not afford them any mindset at all with which to interpret the data. Another potential problem is that the data which users explored was processed and created before we had fully completed the tool, meaning some of the counts and patterns in the trees were inaccurate. However, we decided that this was not a pressing concern, being that we were primarily with whether any interesting reports could be made at all, not with whether these reports were 100% accurate assessments of reality.

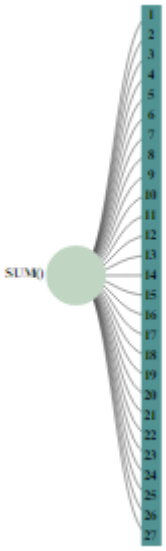
Nevertheless, these user studies were a formative moment in evaluating the tool's performance, generating a number of directions or corrections in the process. We present an overview of responses below, with the four participants recorded simply as P1 through P4.

A. *Positive Responses*

The inclusion of formula examples, grounded in and linking back to the originating spreadsheets, proved to be an especial draw to the tool. After all, none of the participants professed themselves to be experts with Excel, their self-reported proficiencies ranging from "fairly familiar" (P2) to "not extremely familiar" (P4); even if they were experts, it would likely be unreasonable to expect an evenly distributed familiarity with the 100+ function trees available. Nevertheless, the list of actual formulas accessible through a node's double-click allowed the users a way to find examples of formulas with exactly the configuration they wanted. For example, as P1 explored the VLOOKUP function, even though they knew the official API more verbosely explained its signature, they said they could use the tool to collect more examples than the documentation offers.

Furthermore, it offers a simple way of finding examples where certain functions were used together. though a standard text-search tool might yield the same results for a function alone, combinations can require more complex text queries, whereas here, the information is already captured in nodes.

[This paragraph contrasts it with previous approaches, but lacks compelling evidence – a lot of ”mights”]



The discovery of errors and bad design, whether by purpose or accident, produced the more jarring events of the study, and as such, most users commented on the tool's ability to locate these problems in spreadsheet design. This was often supported by the distinctive visual signatures of certain bad practices: multiple arguments produce towers of nodes, deep function nesting stretches the tree horizontally, and so on. As such, the participants remarked on the problems quickly after opening: P3, exploring the example for a 27-argument SUM (with SUM's initial appearance pictured left), diagnosed the situation with either incompetency or just plain weirdness; P2, meanwhile, decided that the spreadsheet's author ”was having a bad day” when writing an AVERAGE with 18 distinct arguments. Aside from these implicit cues, there are also nodes which refer to formula errors, prefixed with #s, which P4 said could be useful if marked well. Furthermore, once these errors are found, the participants could then open the offending spreadsheets, discovering that, even in context, some of these formulas remained incomprehensible, posing problems for mere functioning and maintainability. [Hermanns discussed in one of her paper the transfer scenarios – would go well here.] The tool, then, aggregates these issues in a single place, accessible to either the unguided user and the ones who know exactly where to find these issues.

The detection of bad design, however, is certainly not limited to a visualization tool, and some users remarked that a tool which printed a list of errors could suffice just as well for that purpose. However, they also remarked that this visual interface better supports the case when searching for new and unknown brands of bad smell. Because the exploratory design makes no qualitative judgment and visualizes benign designs with the problematic, it therefore allows for flexible interpretation of what is an issue; that is, a user might not realize a certain design is suboptimal until they see it in the tree and, at the point, find every instance of it.

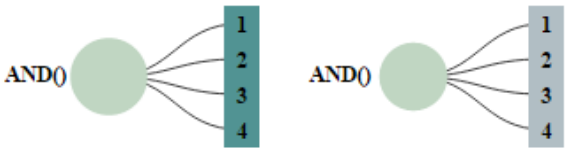
B. Negative Responses

Some interaction with the tool underlined how a lack of contextual information may limit the tool. For example, the benefits described in the previous sections imply a hidden complication: if the tool educates through example but doesn't explicitly distinguish between good design and bad design, how can we be sure that people won't inadvertently learn bad design from it. Right now, we can't be sure; the tool would have to be supplemented with a distinguishing eye, informed on bad design already, and supplying the tool with this would take it beyond a design-agnostic view. [More on this in the conclusion.]

Furthermore, just because a user can exhaustively explore every variation of how a function was used doesn't mean they will learn exactly what it does. P1, in their exploration, examined functions that they hadn't used before, such as KURT and PV, and described precisely what types of arguments it accepted and how many. However, even after exploring the spreadsheets, they could not accurately describe what the functions produced without consulting the official documentation. Whether through reticence of the tool or obscurity of the functions themselves, the exploratory environment is not enough for independent education. However, unlike the previous problem, this can be addressed, perhaps, by guiding the user directly to such documentation from within the tool without violating any design concerns.

A threat to the exploratory philosophy may also mount from the problem of too much data, particularly in the popular functions, like SUM and IF. We had added some precautions before the study: for any position in the tree, for example, only the ten most common functions were displayed without clicking on the expansion arrow ▼. Nevertheless, P3 remarked that they specifically avoided the most popular functions, anticipating a flood of nodes from which they could salvage nothing interesting – which might not have been a bad prediction, considering that the SUM is built from 910 nodes and IF, at least 18000 [double-check these with refined data]. P2 corroborated this by explaining how they gravitated toward specific examples over the plane of nodes but complicates it further by also deeming the tool at its best when it shows either every possibility of an side-by-side (for comparison) or nothing on that branch at all, a balance that is hard to strike with node-dense trees.

Throughout the study, it became apparent that some of the concepts, particularly optional arguments, were easily lost in the representation. To illustrate this, consider the following two trees:



The difference is, perhaps, too subtle: on the left, the tree contains instances of AND functions with any number of arguments, up to four; on the right, the tree contains only instances with exactly four arguments. In other words, an instance of ”=AND(A1, B2, C3, D4)” would be found in both trees, while ”=AND(A1, B2)” would be found in only the left. Though the users naturally did not explicitly report misinterpreting the icons, it became clear in their out-loud thoughts that they viewed each box as a fundamentally different set of possibilities in formula construction rather than just a specific parameter. Though we clarified the meanings when these misunderstandings became apparent, these events point to a failure in the tool to properly indicate every function with clarity.