Faster, Higher, Stronger[1]: Redesigning Spreadsheets for Scale

Mangesh Bendre, Tana Wattanawaroon, Sajjadur Rahman, Kelly Mack,
Yuyang Liu, Shichu Zhu, Yu Lu, Ping-Jing Yang, Xinyan Zhou,
Kevin Chen-Chuan Chang, Karrie Karahalios, Aditya Parameswaran
University of Illinois at Urbana-Champaign (UIUC)
{bendre1, wattana2, srahman7, knmack2, yuyangl2, szhu28, yulu3, py2, xzhou14, kcchang, kkarahal, adityagp}@illinois.edu

***Abstract*—Spreadsheet tools are ubiquitous for interactive ad-hoc data management and analysis. With increasing dataset sizes, spreadsheet tools fall short—they freeze during heavy computation within the sheet (*interactivity*); they are hard to navigate when datasets go beyond a certain size (*navigability*); they only support cell-at-a-time computation, severely limiting analysis capabilities (*expressiveness*). We have been developing DATASPREAD to holistically unify databases and spreadsheets to leverage the benefits of both, with a spreadsheet-like front-end and a database-like backend. We demonstrate three key features of DATASPREAD to address the aforementioned spreadsheet scalability challenges in interactivity, navigability, and expressiveness[1]. Our demonstration will let attendees perform typical analysis tasks on Microsoft Excel and DATASPREAD side-by-side, providing a clear understanding of the improvements offered by DATASPREAD over traditional spreadsheet tools.**

I. INTRODUCTION

Spreadsheet tools are commonly used for interactive analysis and management of data by novice and expert users alike. From personal bookkeeping to complex financial reports, scientific data analysis to scheduling, the ubiquity of spreadsheets as a data management tool is unparalleled with a user base of over 750 million [1]. The main advantages offered by spreadsheet tools include the ability to directly manipulate data, an intuitive user interface, and a flexible data model. However, the sheer scale of data across domains has exposed several limitations of spreadsheet tools while operating on large datasets. Our recent study of scalability issues in spreadsheet tools via analysis of Reddit posts [8] revealed a number of issues with Microsoft Excel that were due to the scale of data or operations. Anecdotal evidence also indicates that users struggle with scalability problems on large and/or complex spreadsheets. To counter these problems, spreadsheet tools curb the size of data being analyzed, *e.g.*, Excel (Google Sheets) imposes a 1M (2M) limit on the number of rows (cells). Even so, many of the Reddit posts indicate scalability issues arising with tens of thousands of rows, well below the 1M row mark [8].

We identify three key scalability limitations of traditional spreadsheet tools using Excel as a typical example. *(a) Interactivity.* Excel ceases to be interactive when dealing with computationally heavy spreadsheets. One Reddit user posted that complex calculations on Excel can take as long as four hours to finish, during which the interface is unresponsive: *"approximately 90% of the time I spend with the spreadsheet is waiting for it to recalculate"*[2]. *(b) Navigability.* While Excel supports basic browsing via scrolling, it does not support rapid navigation to desired spreadsheet areas, or provide a high-level overview. One Reddit user commented: *"inexperienced Excel users are unable to navigate data efficiently"*[2]. *(c) Expressiveness.* Excel uses a cell-at-a-time formula query model—making it cumbersome to express the much more convenient and powerful relational operations when working with tabular data. For example, to filter records that occur between two dates, one user suggested the following formula:

```
IFERROR(INDEX(A:A, SMALL(IF(($A$20:$A$1000
>= $E$40) * ($A$20:$A$1000<=$F$40),ROWS
($A$20:$A$1000)), ROWS ($C$5:$C5))),"")
```

[2], a cumbersome way to express the simple relational operation.

Contributions. To remedy the above limitations and others, since 2014, we have been developing DATASPREAD, a system that *holistically unifies spreadsheets and databases to enable interactive ad-hoc manipulation of large datasets*. DATASPREAD is available as a web application; its source code, documentation, and user guide can be found at http://dataspread.github.io. Our earlier work [3], [4] introduced the feasibility of this unification between a spreadsheet front-end and a database back-end (PostgreSQL), enabling the analysis of datasets exceeding main-memory limitations on a spreadsheet front-end.

In this demonstration, we showcase substantial extensions to our prior prototype, introducing new and novel features aimed at making DATASPREAD interactive, navigable, and expressive.

(a) **Asynchronous computation** addresses the issue of poor interactivity due to slow computation that causes the spreadsheet to hang. Instead, users can have control of the sheet throughout, while cells that require recomputation are processed in the background. Interactivity must be achieved while ensuring consistency, such that no "stale" values are displayed. *Asynchronous computation is displayed in action in Fig. 1, and described in detail in Section II.*

[1] *Citius, Altius, Fortius*, or Faster, Higher, Stronger, is the motto for the Olympics, adopted in 1894 (https://stillmed.olympic.org/Documents/Reports/EN/en_report_268.pdf). Here, we draw a parallel to the three facets of spreadsheets that we are enhancing—interactivity, navigability, and expressiveness.

[2] All Reddit quotes are paraphrased to preserve anonymity.

(b) A **navigation interface** enhances navigability by enabling users to drill-down to desired areas of the data while examining a summarized view. Supporting such an interface requires a data structure that can capture drill-down, roll-up, and scroll operations on the data at various granularities while providing summarized information. *Our navigation interface is displayed in action in Fig. 2, and described in detail in Section III.*

(c) **Table-oriented formulae** improve expressiveness by providing a simple but effective means to express relational operations on tabular regions. For example, the complex formula from earlier can be expressed as a simple table-oriented formula: `SELECT(A20:A1000, AND(ATTR_DATE <= F40, ATTR_DATE >= E40))`. *Table-oriented formulae are displayed in action in Fig. 3, and described in detail in Section IV.*

Related Work. A number of papers have tried to enrich spreadsheets and databases with features from each other, without a holistic integration. Some of these papers have targeted expressiveness. For example, Tyszkiewicz [10] adds SQL capabilities to a spreadsheet, but loses the cell-at-a-time formula semantics of spreadsheets. Similarly, Liu et al. [7] develop a direct manipulation algebra for relational operators, with the same limitation. On the other hand, Witkowski et al. [12] adds formula capabilities to a database, losing the interactive interface. While asynchronous computation has been leveraged in other settings, to the best of our knowledge, it has not been applied to spreadsheets [2]. Similarly, while there has been some work on providing summaries of spreadsheet data [6], no work has provided a multi-granularity navigation interface for spreadsheets. ABC [9] had similar goals to DATASPREAD without addressing the challenges of interactivity, navigability, and expressiveness. Overall, no system has fully addressed the above challenges, while respecting spreadsheet semantics.

Demonstration Overview. Conference attendees will interact with a dataset from Airbnb [5], using DATASPREAD and Excel, to understand the differences in capabilities. This dataset contains information about Airbnb listings (*e.g.*, type, location, price, availability), with ≈570k rows and 15 columns. Our demonstration includes an example of a journalist analyzing these listings, with tasks that emphasize new features we developed to address the shortcomings of traditional spreadsheet tools with respect to interactivity, navigability, and expressiveness, described in the following sections. Attendees are also encouraged to experiment with tasks of their choosing.

II. INTERACTIVITY: ASYNCHRONOUS COMPUTATION

Traditional spreadsheets adopt a *synchronous computation model*. That is, if a change, be it a value or a formula, triggers recomputation, control is only returned to users when the computation is complete; until then, users are kept waiting, possibly for minutes, disrupting activity.

A. Feature Description

DATASPREAD adopts an *asynchronous computation model*, where the cells impacted by the update are recomputed in the background, ensuring *interactivity* by returning control back to users quickly. A naïve method of using asynchrony is to return control to users immediately after an update, leaving all cell values intact. Once a cell is determined to be impacted by the update and has finished recalculation, the value is updated. Such a method can confuse users with inconsistent data, because it is unclear whether the value currently displayed is outdated. Instead, our model ensures *consistency* by quickly detecting cells whose values are stale, and for those cells, we hide the stale values and instead display individual progress bars to indicate the progress of recomputation (Fig. 1b). Overall, this model ensures interactivity by bounding the time for which the system remains unresponsive, while also guaranteeing consistency by preventing the display of stale values.

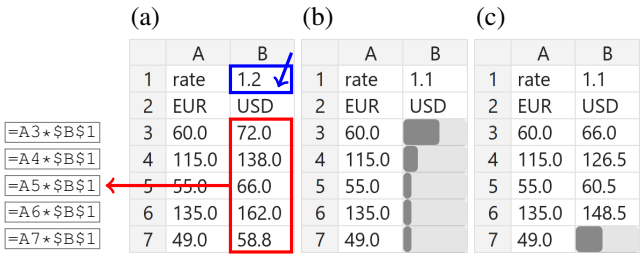

Fig. 1: Asynchronous formula execution. (a) Users modify the conversion rate in cell `B1`. (b) Control is quickly returned, and impacted cells are replaced with progress bars. (c) Each cell value is displayed once computation is complete.

Simultaneously maintaining consistency and interactivity is a challenge. This requires addressing two problems, both of which are NP-HARD. First, the system must identify the impacted cells, those whose values depend on the updated cell(s), in bounded time. Unfortunately, the graph that captures formula dependencies can be very large, making this determination time-consuming. Notice that this graph can tolerate *false positives*, *i.e.*, identifying a region as being impacted by an update, even when it actually is not; false negatives are not permitted, as they violate consistency. Using this insight, we develop a greedy graph compression algorithm that exploits false positive tolerance to quickly identify impacted cells. Second, the system must efficiently schedule the computation of the impacted cells. We implement a novel on-the-fly scheduling algorithm, a variant of weighted shortest-job-first, to compute the impacted cells efficiently in a cache-friendly manner, and prioritize visible cells, thereby minimizing the time that users wait for the values. Experimentally, these algorithms can reduce *unavailability*, the number of unavailable cells (*i.e.*, grey cells in Fig. 1b) over time, by up to 12×, compared to traditional spreadsheets [2].

B. Demonstration Scenario

The following task shows that when there is an update that affects a large number of cells, DATASPREAD remains interactive. Suppose our journalist (from the introduction) has Airbnb listings with rent prices in two currencies, EUR and USD, where the latter is computed using the conversion rate in cell `B1` (Fig. 1a). The journalist wants to update the conversion rate, affecting all of the values in the USD column. On Excel, upon updating the cell, the interface will be unresponsive

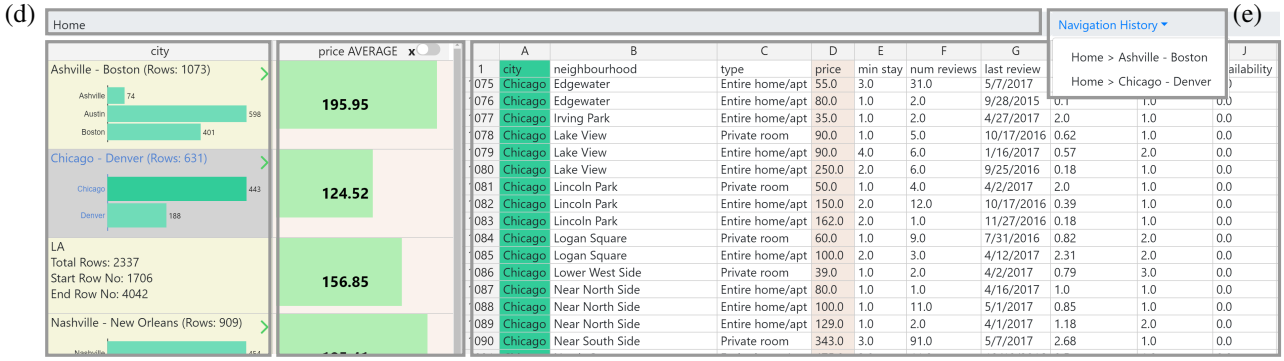

Fig. 2: Navigation Panel. (a) hierarchical overview, (b) aggregation results, (c) spreadsheet, (d) navigation context, and (e) navigation history.

until the computation is complete. In contrary, the attendees can perform this update in DATASPREAD interactively; *i.e.*, control is returned to users almost immediately. The cells impacted will not show outdated values. Instead, they will display progress bars (Fig. 1b). A correct value will be shown once the background recomputation for that cell is complete (Fig. 1c). Attendees can also write their own functions, possibly more complicated or computationally intensive, and witness how the system remains interactive.

III. NAVIGABILITY: SEAMLESS NAVIGATION

Spreadsheets adopt scrolling for accessing and displaying data that does not entirely fit in a screen. However, when users scroll through a dataset with thousands of rows, they can easily lose context [11]. Built-in spreadsheet features like pivot table and external tools like Tableau or PowerBI address some of these challenges. However, these tools do not exist in-situ with the raw data and the supported analysis operations (*e.g.*, in pivot table) are limited. As a result, users are forced to alternate between tools to navigate and analyze data.

A. Feature Description

We propose a zoomable overview interface that is integrated as a data exploration plugin for DATASPREAD. The interface abstracts the data to *levels* of varying granularity, akin to online maps, *e.g.*, Google Maps. At each level, the spreadsheet data is abstracted as a group of bins—the grouping is essentially an equi-depth histogram constructed on the attribute by which the data is organized (Fig. 2a). Each bin contains high level summary (aggregated information) of the data/region it spans. The bins at any given level are further grouped to form the next higher level of granularity, effectively forming a hierarchy of bins. We now explain the features supported by the interface.

Navigational Operations. Users can navigate the data by selecting any bin and jump to the corresponding location within the spreadsheet. For example, when users select the *Chicago-Denver* bin, the spreadsheet displays the first 30 Chicago listings (Fig. 2c) while the bin is highlighted (Fig. 2a). If users scroll to a different location in the spreadsheet, the corresponding bin on the overview is also highlighted. As the overview interface is hierarchical, users can drill-down (or zoom) into a specific bin or roll-up to (or zoom out of) a different level in the overview. For example, if users drill-down into the *Chicago-Denver* bin by clicking on it, the overview will display three cities Chicago, Dallas, and Denver in detail.

Analytical Operations. Users can issue traditional spreadsheet aggregation formulae on the overview bins and view the results as either raw values or charts. As users zoom in and out of the overview interface, the results continue to be updated as the hierarchy changes. Users can choose to perform different analysis operations on the overview simultaneously. For example, as users choose to compute the average price of listings in different bins, a new column is added that displays the aggregated result per bin (Fig. 2b).

Contextual and Historical Information. The interface maintains the current navigation context through a breadcrumb-like feature (Fig. 2d), along with the history of visited bins so that users can move back and forth (Fig. 2e).

The primary challenge in building this interface to seamlessly integrate spreadsheets with the overview interface and facilitate rapid interactive exploration and drill-down/roll-up. In addition, we need a data structure that satisfies the requirements of such an interactive navigation interface, *e.g.*, for dynamic reordering and summarization of data. To enable seamless integration of the navigation interface with the spreadsheet data, we leverage our work on hierarchical positional indexes [4]. Each level in the positional index maps to a corresponding level in the navigation interface. This index needs to be reconstructed every time the data is reordered, and we display approximate partial results to maximize interactivity. We leverage DATASPREAD's formula engine to generate summary statistics (Fig. 2b). One issue is that supporting multiple summary requests (*e.g.*, `AVERAGE`, `COUNT`) requires multiple scans. We exploit sharing of computation on the subsets of data for each bin.

B. Demonstration Scenario

The following task shows that users can quickly navigate through a large spreadsheet and gain high level insights. Suppose our journalist wants to compare the average price of the listings in Chicago and Los Angeles (LA), and find the availability of the three cheapest listings in both cities. With Excel, the attendees have to first sort the data by "City", filter out the Chicago listings, and finally, compute the average by providing the range of the Chicago listings to the `AVERAGE` function. Then, attendees have to sort the Chicago listings in descending order of price and find the availability of the three

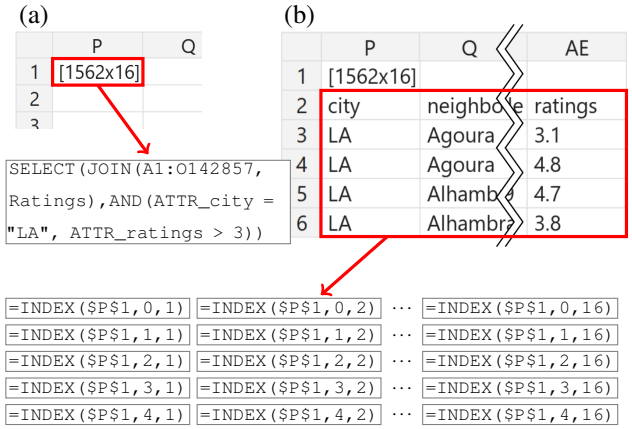

Fig. 3: JOIN operation in DATASPREAD. (a) Users write the JOIN formula. (b) Users then retrieve JOIN results by writing the INDEX formula in P2 and using autofill for the rest.

cheapest listings. They have to repeat the same process for LA. Attendees will witness how tedious this approach can be for large spreadsheets as it involves scrolling endlessly to find the range of the Chicago (LA) listings. As explained earlier, using the features provided by DATASPREAD, users can effortlessly accomplish these tasks. Attendees will also be able to use the navigation interface to freely zoom in/out and scroll through various granularities of the data.

IV. EXPRESSIVENESS: TABLE ORIENTED FORMULAE

Present spreadsheet tools do not support computation that go beyond the cell-at-a-time metaphor: for example, relational algebra operations such as general joins are not permitted or at least not straightforward (*e.g.*, VLOOKUPs for key-foreign key joins). DATASPREAD supports both SQL as well as general-purpose relational computation via *table-oriented formulae*, supporting operations such as joins, on both tables from the underlying database or spreadsheet regions, *e.g.*, A3:D4, which are treated as tables. Table-oriented formulae enable the use of relational primitives in the same way as other formulae.

A. Feature Description

We support the following table-oriented functions: UNION, DIFFERENCE, INTERSECTION, CROSSPRODUCT, JOIN, SELECT, PROJECT, and RENAME. These functions return a single value of type *composite table*, representing the tabular result of an operation. Unlike *e.g.*, integers or strings, a cell holding composite table values does not show the complete value (as a table), but instead only shows the table dimensions, *e.g.*, [3x4]. The composite table type allows for nesting to obtain complex expressions, *e.g.*, UNION(T_1,UNION(T_2,T_3)) for a union of three tables. In addition, users can issue SQL queries using the SQL function, which is directly executed by the underlying database, and obtain a composite table result. The INDEX(*cell*,i,j) function returns the value of the i-row and j-column in the composite table at *cell*. To display a complete composite table value, users can create an array of cells using INDEX calls for various (i, j) values by first entering the INDEX formula in one cell, and then complete the rest automatically by dragging the autofill handle. The special value $i = 0$ provides the header row (Fig. 3b).

The primary challenge is to ensure that the table-oriented formulae work seamlessly with the cell-at-a-time formula model of Excel in spite of the differences in semantics. Specifically, a table-oriented formula can return multiple records, making it incompatible with spreadsheet formulae that evaluate to a single cell. Returning multiple records is also problematic because unless we are careful, these records can overflow and overwrite other data. The use of INDEX to look up entries in the composite table value avoids the issue of overflow of records, since only the cells that use an INDEX formula will ever refer to data corresponding to the tabular result.

B. Demonstration Scenario

The following task shows that operations such as JOIN and SELECT can be expressed simply, in a manner consistent with typical spreadsheet formulae. Suppose, in addition to the original spreadsheet of Airbnb listings, the journalist finds a "Ratings" spreadsheet that provides, for each Airbnb listing (given by its ID), its (average) ratings. The journalist wants to see details of all the listings in LA above some ratings threshold. The task corresponds to a *natural join* plus a filter, which is not possible natively in Excel. The closest solution is an *outer join*, done by a combination of VLOOKUP and IF statements which must then be applied to multiple listings by dragging the autofill handle. Another approach is to use Excel's filtering capabilities after a VLOOKUP, but that will impact other data on the sheet. Using DATASPREAD, the attendees can use the following formula in cell P1: SELECT(JOIN(A1:O142857, Ratings), AND(ATTR_city = "LA", ATTR_ratings > 3)). Cell P1 displays the dimensions of the result table (Fig. 3a). The attendees can use INDEX($P$1,i,j) to view the result table itself, starting with $i = 0$ and $j = 1$ and dragging the autofill handle for different i and j values (Fig. 3b). Attendees can amend the query and see how the result cells adapt to the new query without overwriting other cells that already contain data.

REFERENCES

[1] How finance leaders can drive performance. https://enterprise.microsoft.com/en-gb/articles/roles/finance-leader/how-finance-leaders-can-drive-performance/.
[2] M. Bendre et al. Anti-freeze for large and complex spreadsheets. Technical Report. http://dataspread.github.io/papers/async-formulae.pdf.
[3] M. Bendre et al. Dataspread: Unifying databases and spreadsheets. volume 8, pages 2000–2003. VLDB Endowment, Aug. 2015.
[4] M. Bendre et al. Towards a holistic integration of spreadsheets with databases: A scalable storage engine. In *ICDE*, April 2018.
[5] M. Cox. Inside Airbnb. http://insideairbnb.com/get-the-data.html.
[6] M. Joglekar et al. Smart drill-down. *PVLDB*, 2015.
[7] B. Liu and H. Jagadish. A spreadsheet algebra for a direct data manipulation query interface. In *ICDE*, pages 417–428. IEEE, 2009.
[8] K. Mack et al. Characterizing scalability issues in spreadsheet software using online forums. In *SIGCHI*, 2018.
[9] V. Raman et al. Scalable spreadsheets for interactive data analysis. In *ACM SIGMOD Workshop on DMKD*, 1999.
[10] J. Tyszkiewicz. Spreadsheet as a relational database engine. In *SIGMOD*, pages 195–206. ACM, 2010.
[11] J. Watts-Perotti et al. How experienced users avoid getting lost in large display networks. *International J. of HCI*, 11(4):269–299, 1999.
[12] A. Witkowski et al. Query by excel. In *VLDB*, pages 1204–1215. VLDB Endowment, 2005.