Summary

This whitepaper lays out the technical design space for delivering Anyscale Interactivity—the key to enabling end users to wrangle data at any scale. First we consider the fundamental limits facing algorithms and provisioning. Then we focus on the key issue: keeping the working set size within real-time interactivity constraints. Finally we review how working set size constraints for data wrangling can be met intelligently via a suite of sampling techniques.

Background

The last few years have seen the emergence of a new set of data wrangling products that allow end users to assess, explore, transform, blend and clean data. Many of these tasks were traditionally performed by programmers or IT staff, who used technical ETL products as part of long-term data warehousing design cycles. In modern contexts, this traditional approach is inefficient or even infeasible: IT staff and their tool chain become a bottleneck in the data supply chain. Instead, the end-users of the data—the ones who are close to the use cases—are adopting intelligent visual data wrangling solutions that enable them to perform rapid self-service exploration, prototyping and even long-term deployment.

End-user data wrangling software needs to be designed carefully to give users the power to work with data at any scale. Interactive performance becomes quite challenging in that context.

Design Requirements

Technically, effective data wrangling solutions present two significant performance challenges: the need to ensure a rich and responsive user interface (UI), and the need to scale up and down with data sizes.

UI Responsiveness

Interactive visual performance is an absolutely essential aspect of effective data wrangling. Research studies show that UI delays beyond 500 milliseconds impair user behavior in critical ways—users decrease their activity and examine less of the data, which in turn leads to decreased rates of more fundamental tasks like observation, generalization and hypothesis. In short, even modest delays in user experience lead to poor data quality and decreased insight.¹

Data Scale

At the same time, a reality of modern technology is that data sizes are increasingly diverse. Sometimes the most important information an analyst needs may be contained in a desktop spreadsheet, where 500ms response is taken for granted. Other times, it may spread across the records of a massive log file, where the latest Big Data solutions struggle to respond in minutes. This diversity of scale is no longer confined to the IT department—end-users in various lines of business need to be able to work with data at a variety of scales.

Traditionally, products and roles were segmented to solve one of these challenges at the expense of the other. End-users were given responsive tools like Excel and SAS, which allow fast interaction with data by

Based on numerous research studies, the average attention span for someone interacting with a software product is small. Response times need to be fast enough so the person using the product is always in the flow of their work. Here is a breakdown of how different response times impact users:

- 0.1 sec: limit for user perception of “instantaneous”
- 0.5 sec: limit for user flow of thought
- 10 sec: limit for keeping user attention

enforcing limits on data size. IT users were given batch-processing tools like ETL solutions, which provide technical user interfaces that focus on displaying data descriptions (schemas) rather than data. From a process perspective, this meant that (a) end-users were at the mercy of IT staff to prepare “cuts” of the data that were small enough to fit into the end-user tools, and (b) IT staff were required to spend time shaping and scaling down data to the (often vaguely-specified) needs of end users.

Self-service analytics requires breaking this cycle via solutions that can provide Anyscale Interactivity: a responsive end-user experience for any scale of data.

**Technical Approaches to Anyscale Interactivity**

There are a few basic technical approaches to providing interactivity at any scale. In the end, they all must respect some basic arithmetic of computing: the cost of handling a dataset goes up with the size of the dataset. That cost can be shifted to different points in time or different tasks, but eventually it has to be paid. Here we review the three basic approaches to Anyscale Interactivity.

1. **The Expensive Solution: Provision Compute to Match Data**

A common way to ensure 500ms response time is to guarantee that there is enough compute and near-line memory to touch every data item within 500ms. Spreadsheets achieve this by abandoning the anyscale requirement: they limit file size to the CPU and RAM capacity of a typical laptop. To achieve anyscale capabilities, you need to provision a cluster (or prepare for cloud computing bills) with enough cores and RAM to provide interactive response on the largest volume of data your entire user community will want to wrangle at any given time.
This might appear to be the gold-plated “no compromises” approach to anyscale interactivity: the cost is paid explicitly in hardware resources. But in practice this approach is often not truly anyscale—any well-run organization caps costs somewhere, which requires capping scale as well. This is typically the case for Software-as-a-Service providers, or large-scale deployments. Hence this approach is brittle: performance is acceptable until the scale cap, at which point it degrades to the point of unusability.

2. The OLAP Solution: “Pre-process” Data Offline
In specialized settings, it is possible to preprocess the data and build summary structures (indexes, rollups, materialized views, etc.) The cost of preprocessing is paid up front before usage, in an offline batch mode using modest hardware combinations of CPU, RAM and workspace disks. Afterward, users enjoy interactivity, because the user experience works directly off the summary structures without touching the original data. This idea powers many Business Intelligence (BI) and “OLAP” solutions, where it is a good fit: these tools provide a fixed set of user actions on a groomed dataset.

Unfortunately, “pre-processing” is not an option for data wrangling. The majority of use cases for data wrangling require multi-step transformations to the data. Many of those transformations produce a result that has different structure and values than the input. Whenever the structures or values change, the result has to be “pre-processed” again. In short, the offline “pre-processing” requirement becomes an online during-processing requirement, and interactivity is lost. In addition, many exploratory data wrangling tasks begin with raw files that have yet to be “pre-processed” at all—precisely because they need to be wrangled first. Like the provisioning approach, the “pre-processing” approach is brittle for data wrangling use cases: even if it works at first, it quickly degrades to an unusable, non-interactive experience.

3. The General Solution: Recipes on Working Sets
This final approach is a twist on the first: rather than provisioning your compute infrastructure to the maximum data size, the software extracts a working set of data that is right-sized to the computing capacity allocated to the problem—whether that is on an end-user desktop or in a cluster. End-users do agile exploration and prototyping on a working set of data, and generate a repeatable data wrangling recipe. If the dataset is small enough to fit within the working set size (working set = dataset), then the task is complete. For datasets larger than the working set size (working set < dataset), once the recipe looks good on the working set it can be launched in batch mode on the full dataset. The cost is paid after the wrangling recipe is prototyped—the batch job produces a completely-wrangled output, including an assessment of the resulting data quality.

The working-set/recipe approach is the only one that provides true anyscale functionality with an interactive user experience. A key challenge is raised by this approach: for datasets larger than the working set size, the software needs to extract a working set that leads users to write recipes that generalize well to the full dataset. Data sampling provides the technical methodology to extract working sets intelligently.

Because the third approach is the only one that is truly anyscale, you should exercise caution when evaluating claims to address interactive wrangling using the first two approaches. With that in mind, the next challenge is to evaluate the flexibility of the sampling approaches used to achieve the third solution.
Approach

1. Provision resources to match data usage

2. “Pre-process” data offline

3. Recipes on sample working sets

Wrangling Responsiveness

<table>
<thead>
<tr>
<th>High</th>
</tr>
</thead>
<tbody>
<tr>
<td>High scales with number of concurrent users and data volume</td>
</tr>
<tr>
<td>Low</td>
</tr>
<tr>
<td>data requires additional process when iterating on complex transformations</td>
</tr>
<tr>
<td>High</td>
</tr>
<tr>
<td>wrangle in real time on working sets</td>
</tr>
</tbody>
</table>

Resource Consumption

<table>
<thead>
<tr>
<th>Low</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low</td>
</tr>
<tr>
<td>Low extract necessary working sets from full dataset</td>
</tr>
</tbody>
</table>

Approaches to Sampling

Guiding Principles
When extracting informative working sets from large datasets, a good data wrangling solution should simultaneously ensure user agility, provide rich feedback, and surface relevant data attributes. Here are a few guiding principles:

Quick Start
Even for enormous files, the user interface should be immediately populated. The initial working set should surface first-order characteristics of the data including schema and structure, allowing users to jump headfirst into exploration and wrangling rather than having to first wait for a complex working set to be extracted.
Automated Refinement
As the user is examining a “quick-and-dirty” working set, backend processing should surface a more nuanced working set that better represents the totality of the data. This helps assure the user that the recipe they are crafting is robust, covering the full range of values and representative summary statistics, as well as outliers and anomalies in both structure and value. As this refined working set becomes ready, the user interface should notify the user that they can switch their view to the refined working set.

User Refinement
In some cases, users will want to explicitly adjust the sampling strategy to focus on specific aspects of the dataset, and ensure that their working set represents that data well. To support this kind of user control, sampling should be a first class part of the user experience, and offer a variety of easily understandable configurations to fulfill user needs.

Re-sampling
As users compose increasingly complex recipes that blend (join, union etc.), summarize (aggregate, group, ...) and reshape (pivot, flatten, unnest...) data, the structure and granularity of the results may be very different than in the input data. After these steps, new working sets should be surfaced from the transformed results, to provide accurate feedback on the output of the recipe steps. User control over sampling should be available every step of the way, and new samples should be automatically suggested when changes to the recipe are made.

Sampling Strategies: What’s in a Working Set?
In a system that extracts several working sets, users are free to explore different characteristics of the underlying dataset. Certain working sets might highlight structure; others might summarize statistics and find outliers; others might surface specific values or attributes within the data. To provide a flexible sampling solution that satisfies a wide range of needs, users must be able to achieve a variety of goals via different sampling strategies.

First Rows Samples for Quick Start and Structure:
To effectively transform datasets, the structure of the data must be understood. To quickly understand structure, a first rows sample can be extracted with the first \( n \) bytes of a file or the first \( m \) rows of a database table. This type of sample captures information on structure and can quickly expose problems with data types or column and row structuring. First rows samples support our quick start principle, but they can suffer what statisticians call “the Alabama problem”—in a file sorted by state, there will be an unusual bias toward Alabama. That’s one reason why the remaining sampling techniques are worth running in addition; they take more time, but can be prepared in the background while the end-user works with the first rows sample.
Random Samples for Statistics and Visualization:
In most cases, users will want an overview of the trends in the full dataset. It’s often helpful to quickly see a visual depiction of the “shape” of the value distributions: whether it’s bell-shaped, skewed to one side, or more complicated. In a bit more detail it’s handy to know the “typical” values (average, median), the “spread” (standard deviation, quantiles), and the outliers. To surface summary features of the full dataset, a random sample can be computed. Random samples are critical for summary visualizations, which in turn can help make decisions about the right steps to take to clean up data values.

Stratified Samples for Second-Order Transformations:
Certain transformations craft the shape and labels of the output based on the set of all distinct values in the input. These transformations are sometimes called “second-order” transformations, and they include pivots, unnesting of nested formats like JSON and XML. Grouping and aggregation fit into this category as well. In order for the working set to model the final result well, a stratified sample can be extracted—stratified samples include representative records with each of the unique values. This ensures that the result of the working set provides examples of every resulting row and column.

Anomaly Samples for Data Quality:
At some stages of the wrangling process, users do not want their attention on the “typical” data in a random sample; they want to zoom in on “atypical” data that needs to be cleaned up. This often arises when importing data from an unreliable source, or assessing the output of complex recipe steps. An anomaly sample allows users to quickly focus on rows with poor quality, by making sure to include a sizable representation of items with missing, mismatched or outlying values across columns.

Filter-based Samples for User-Driven Exploration:
Sometimes, users may want to focus specifically on classes of items that they know should be in a dataset. For example, if a user is working on a specific category in a dataset known to have pathological behavior, she may want to narrow down her working set to the records in that category before crafting transformations. For these scenarios, filter-based samples allow users to craft specific filter conditions which drill down into the data.
Cluster-based Samples for Compound Data:
Alternately, rather than finding specific values, users may seek to see several complete clusters of records within the data, each cluster corresponding to a compound event. For example, when a user computes metrics across user sessions, she may want to extract a few sessions (each composed of multiple records) and work on those sessions to ensure that her transformations and the subsequently derived metrics make sense. When the meaning of the data requires examining compound groups or clusters of records, cluster-based samples provide a good working set.

Evaluating Data Wrangling Solutions
Modern data wrangling requires addressing end-user interactivity as well as data scale. Anyscale Interactivity is a key requirement, and should be one of the very first issues to assess when evaluating a data wrangling solution. Basic questions should include:

1. Does the software provide interactive previews of data transformations?
2. How does the software handle data that exceeds memory capacity?
3. If the software employs samples, what sampling methods does it offer to users?

As discussed, Anyscale Interactivity is only feasible in a solution that forms working sets that fit in memory. For big data, sampling is the robust way to form working sets. But “sampling” can mean a lot of things. Effective data wrangling solutions should provide a wide range of sampling strategies to support a variety of concerns, allowing the UI and the user to trade off responsiveness, statistical representation, and user-driven exploration and understanding of the data.

About Trifacta
Trifacta, the leading data wrangling solution for exploratory analytics, significantly enhances the value of an enterprise’s big data by enabling users to easily transform and enrich raw, complex data into clean and structured formats for analysis. Leveraging decades of innovative work in human-computer interaction, scalable data management and machine learning, Trifacta’s unique technology creates a partnership between user and machine, with each side learning from the other and becoming smarter with experience. Trifacta is backed by Accel Partners, Greylock Partners and Ignition Partners.

For Additional Questions, Contact Trifacta
www.trifacta.com | 844.332.2821

Experience the Power of Data Wrangling Today
www.trifacta.com/start-wrangling