Systems for Interactive Data Analysis

CSC 630

Streaming and Sample-based systems

So Far

- Data Cubes as a general strategy for fast, rich, interactive exploration

- We’re about 30% of the way through the papers

Couple of Announcements

- I want your feedback! Please fill the form out and return to me.
  - **This is your best chance to influence the future of the course**

Writing Assignment 1

- Choose 2 of the following 4 papers we went over in class:
  - Partial lattice precomputation, Dwarf, Nanocubes, immens
- Write (about) 500 words on each of the two.
  - Think of it as an extended summary of the technique
  - Contributions, shortcomings, connections to previous work, etc.
  - Connection to your potential research project, if any

Why Streaming?

- Do we really need all the data?

Why not streaming?

- What about outliers?

System Papers

Online Aggregation

Joseph M. Hellerstein
Computer Science Division
University of California, Berkeley
jmh@cs.berkeley.edu

Peter J. Haas
Almaden Research Center
IBM Research Division
peterh@almaden.ibm.com

Helen J. Wang
Computer Science Division
University of California, Berkeley
helenjw@cs.berkeley.edu

Abstract

Aggregation in traditional database systems is performed in batch mode: a query is submitted, the system processes a large volume of data over a long period of time, and, eventually, the final answer is returned. This archaic approach is frustrating to users and has been abandoned in most other areas of computing. In this paper we propose a new *online aggregation* interface that permits users to both observe the progress of their aggregation queries and control execution on the fly. After outlining usability and performance requirements for a system supporting online aggregation, we present a suite of techniques that extend a database system to meet these requirements. These include methods for

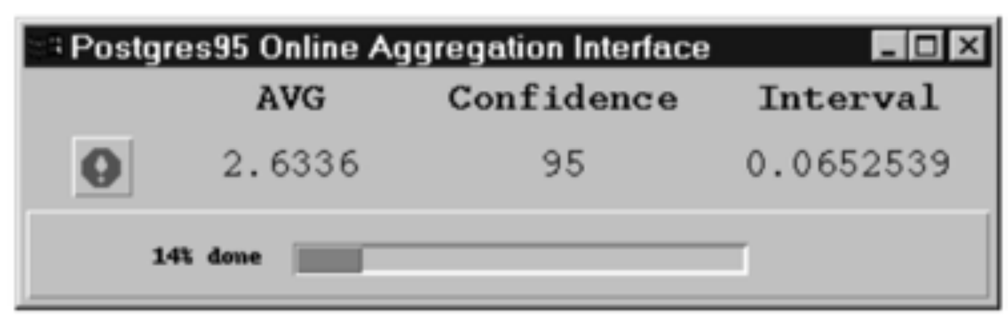


Figure 1: An online aggregation interface for Query 1.

approximation might be available very quickly.

We propose changing the interface to aggregation processing and, by extension, changing aggregation processing

System Papers

Spark: Cluster Computing with Working Sets

Matei Zaharia, Mosharaf Chowdhury, Michael J. Franklin, Scott Shenker, Ion Stoica
University of California, Berkeley

Abstract

MapReduce and its variants have been highly successful in implementing large-scale data-intensive applications on commodity clusters. However, most of these systems are built around an acyclic data flow model that is not suitable for other popular applications. This paper focuses on one such class of applications: those that reuse a working set of data across multiple parallel operations. This includes many iterative machine learning algorithms, as well as interactive data analysis tools. We propose a new framework called Spark that supports these applications while retaining the scalability and fault tolerance of MapReduce. To achieve these goals, Spark introduces an abstraction called resilient distributed datasets (RDDs). An RDD is a read-only collection of objects partitioned across a set of machines that can be rebuilt if a partition is lost. Spark can outperform Hadoop by 10x in iterative machine learning jobs, and can be used to interactively query a 39 GB dataset with sub-second response time.

1 Introduction

MapReduce/Dryad job, each job must reload the data from disk, incurring a significant performance penalty.

- **Interactive analytics**: Hadoop is often used to run ad-hoc exploratory queries on large datasets, through SQL interfaces such as Pig [21] and Hive [1]. Ideally, a user would be able to load a dataset of interest into memory across a number of machines and query it repeatedly. However, with Hadoop, each query incurs significant latency (tens of seconds) because it runs as a separate MapReduce job and reads data from disk.

This paper presents a new cluster computing framework called Spark, which supports applications with working sets while providing similar scalability and fault tolerance properties to MapReduce.

The main abstraction in Spark is that of a *resilient distributed dataset* (RDD), which represents a read-only collection of objects partitioned across a set of machines that can be rebuilt if a partition is lost. Users can explicitly cache an RDD in memory across machines and reuse it in multiple MapReduce-like *parallel operations*. RDDs

System Papers

BlinkDB: Queries with Bounded Errors and Bounded Response Times on Very Large Data

Sameer Agarwal†, Barzan Mozafari°, Aurojit Panda†, Henry Milner†, Samuel Madden°, Ion Stoica*†

†*University of California, Berkeley* °*Massachusetts Institute of Technology* **Conviva Inc.*

{sameerag, apanda, henrym, istoica}@cs.berkeley.edu, {barzan, madden}@csail.mit.edu

Abstract

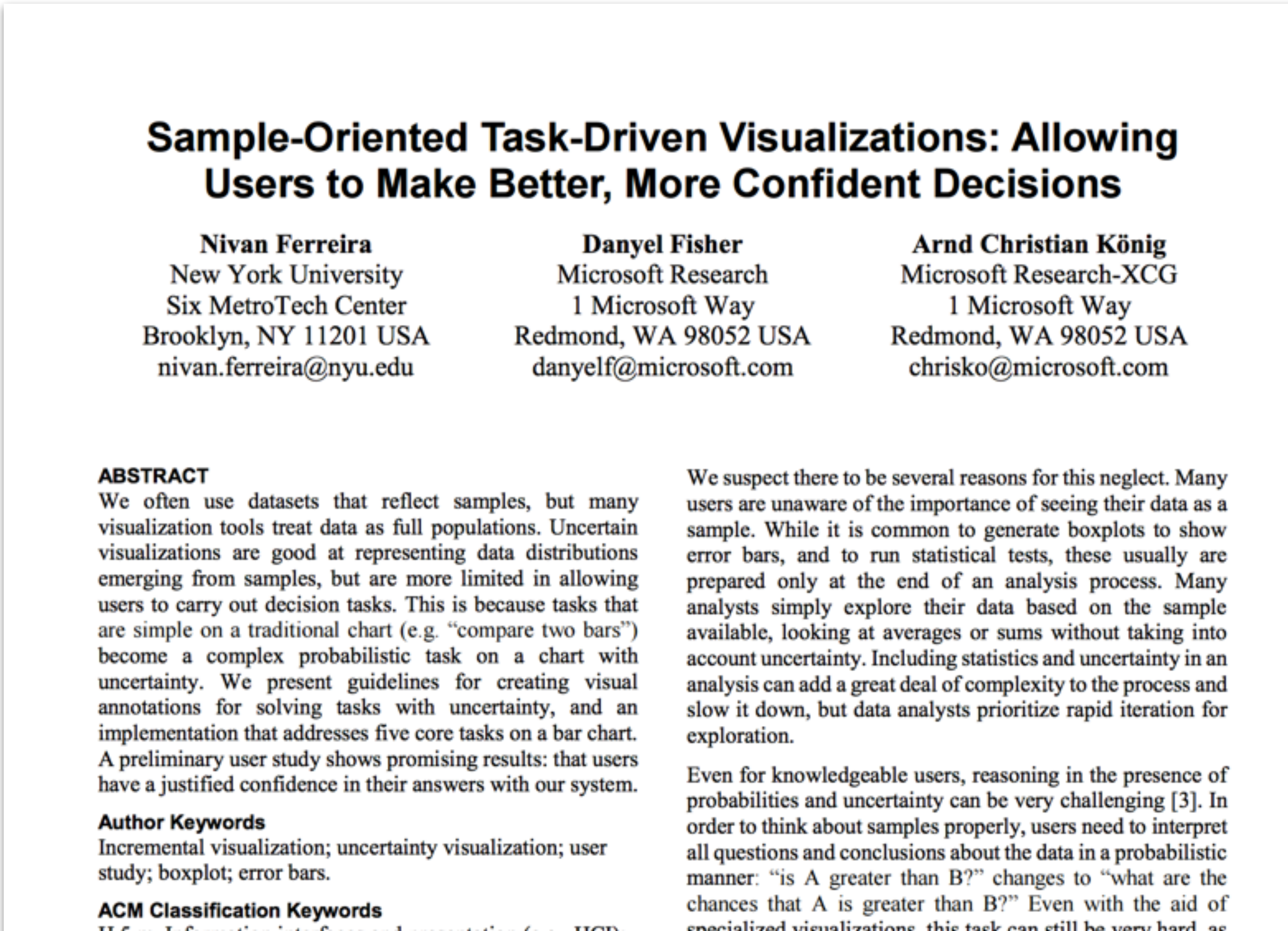
In this paper, we present BlinkDB, a massively parallel, approximate query engine for running interactive SQL queries on large volumes of data. BlinkDB allows users to trade-off query accuracy for response time, enabling interactive queries over massive data by running queries on data samples and presenting results annotated with meaningful error bars. To achieve this, BlinkDB uses two key ideas: (1) an adaptive optimization framework that builds and maintains a set of multi-dimensional stratified samples from original data over time, and (2) a dynamic sample selection strategy that selects

cessing of large amounts of data by trading result accuracy for response time and space. These techniques include sampling [10, 14], sketches [12], and on-line aggregation [15]. To illustrate the utility of such techniques, consider the following simple query that computes the average `SessionTime` over all users originating in `New York`:

```
SELECT AVG(SessionTime)
FROM Sessions
WHERE City = 'New York'
```

Suppose the `Sessions` table contains 100 million tuples for

Interactive Requirements



Sample-Oriented Task-Driven Visualizations: Allowing Users to Make Better, More Confident Decisions

Nivan Ferreira
New York University
Six MetroTech Center
Brooklyn, NY 11201 USA
nivan.ferreira@nyu.edu

Danyel Fisher
Microsoft Research
1 Microsoft Way
Redmond, WA 98052 USA
danyelf@microsoft.com

Arnd Christian König
Microsoft Research-XCG
1 Microsoft Way
Redmond, WA 98052 USA
chrisko@microsoft.com

ABSTRACT
We often use datasets that reflect samples, but many visualization tools treat data as full populations. Uncertain visualizations are good at representing data distributions emerging from samples, but are more limited in allowing users to carry out decision tasks. This is because tasks that are simple on a traditional chart (e.g. "compare two bars") become a complex probabilistic task on a chart with uncertainty. We present guidelines for creating visual annotations for solving tasks with uncertainty, and an implementation that addresses five core tasks on a bar chart. A preliminary user study shows promising results: that users have a justified confidence in their answers with our system.

Author Keywords
Incremental visualization; uncertainty visualization; user study; boxplot; error bars.

ACM Classification Keywords

We suspect there to be several reasons for this neglect. Many users are unaware of the importance of seeing their data as a sample. While it is common to generate boxplots to show error bars, and to run statistical tests, these usually are prepared only at the end of an analysis process. Many analysts simply explore their data based on the sample available, looking at averages or sums without taking into account uncertainty. Including statistics and uncertainty in an analysis can add a great deal of complexity to the process and slow it down, but data analysts prioritize rapid iteration for exploration.

Even for knowledgeable users, reasoning in the presence of probabilities and uncertainty can be very challenging [3]. In order to think about samples properly, users need to interpret all questions and conclusions about the data in a probabilistic manner: "is A greater than B?" changes to "what are the chances that A is greater than B?" Even with the aid of

Interactive Requirements

Trust Me, I'm Partially Right: Incremental Visualization Lets Analysts Explore Large Datasets Faster

Danyel Fisher[*], Igor Popov[†], Steven M. Drucker[*], mc schraefel[†]

[*]Microsoft Research
1 Microsoft Way,
Redmond, WA USA
{danyelf, sdrucker}@microsoft.com

[†]Electronics and Computer Science
University of Southampton
Southampton, Hampshire, UK SO17 1BJ
{mc+w, ip2go9}@ecs.soton.ac.uk

ABSTRACT

Queries over large scale (petabyte) data bases often mean waiting overnight for a result to come back. Scale costs time. Such time also means that potential avenues of exploration are ignored because the costs are perceived to be too high to run or even propose them. With *sampleAction* we have explored whether interaction techniques to present query results running over only incremental samples can be presented as sufficiently trustworthy for analysts both to make closer to real time decisions about their queries and to be more exploratory in their questions of the data. Our work with three teams of analysts suggests that we can indeed accelerate and open up the query process with such incremental visualizations.

Author Keywords

Incremental visualizations, large data, exploratory data analysis, online aggregation.

ACM Classification Keywords

H.5.2. Information interfaces and presentation: Miscellaneous. H.2.8. Database Applications: Data mining.

query costs in a variety of ways. Strategies to accelerate large scale data processing are represented in systems like Dremel [9] and C-Store [18] that churn through large collections of data by pre-structuring the data and moving the computation closer to the data.

So while computational and storage approaches make large scale queries possible, they still often restrict either the number and types of queries that might be run, or avenues that might be explored because the queries must be designed with such care to be worth the wait and the cost of queuing for the resource.

One possible technique, proposed by Hellerstein and others [7], is to query databases incrementally, looking at ever-larger segments of the dataset. These samples can be used to extrapolate estimated final values and the degree of certainty of the estimate. The analyst would get a response quickly by considering a large, initially unclear range of values that rapidly converge to more precise values. This approach may let an analyst iterate on a query with substantially decreased delay and increased flexibility; if

Interactive Requirements

Rapid Sampling for Visualizations with Ordering Guarantees

Eric Blais
MIT
eblais@csail.mit.edu

Albert Kim
MIT
alkim@csail.mit.edu

Aditya Parameswaran
MIT and Illinois (UIUC)
adityagp@illinois.edu

Piotr Indyk
MIT
indyk@mit.edu

Sam Madden
MIT
madden@csail.mit.edu

Ronitt Rubinfeld
MIT and Tel Aviv University
ronitt@csail.mit.edu

ABSTRACT

Visualizations are frequently used as a means to understand trends and gather insights from datasets, but often take a long time to generate. In this paper, we focus on the problem of *rapidly generating approximate visualizations while preserving crucial visual properties of interest to analysts*. Our primary focus will be on sampling algorithms that preserve the visual property of *ordering*; our techniques will also apply to some other visual properties. For instance, our algorithms can be used to generate an approximate visualization of a bar chart very rapidly, where the comparisons between any two bars are correct. We formally show that our sampling algorithms are generally applicable and provably optimal in theory, in that they do not take more samples than necessary to generate the visualizations with ordering guarantees. They also work well in practice, correctly ordering output groups while taking order of magnitudes fewer samples and much less time than conventional

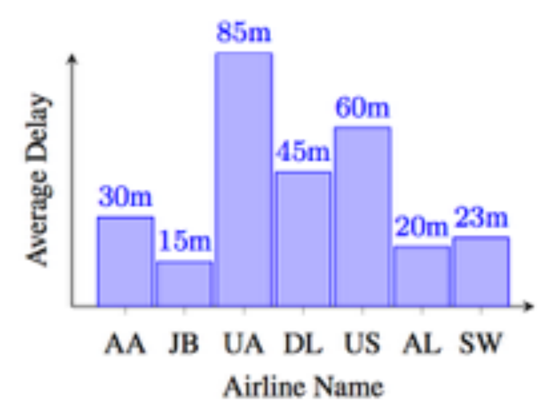


Figure 1: Flight Delays

alization appears similar to the same visualization computed on the entire database. The primary visual property we consider in this paper is the *correct ordering property*: ensuring that the groups or bars in a visualization or result set are ordered correctly, even if the actual value of the group differs from the value that would result if the entire database were sampled. For example, if the delay of

Interactive Requirements

The Case for Data Visualization Management Systems [Vision Paper]

Eugene Wu
sirrice@csail.mit.edu

Leilani Battle
leilani@csail.mit.edu

Samuel R. Madden
madden@csail.mit.edu

Most visualizations today are produced by retrieving data from a database and using a specialized visualization tool to render it. This decoupled approach results in significant duplication of functionality, such as aggregation and filters, and misses tremendous opportunities for cross-layer optimizations. In this paper, we present the case for an integrated Data Visualization Management System (DVMS) based on a declarative visualization language that fully compiles the end-to-end visualization pipeline into a set of relational algebra queries. Thus the DVMS can be both *expressive* via the visualization language, and *performant* by leveraging traditional and visualization-specific optimizations to scale interactive visualizations to massive datasets.

1. INTRODUCTION

The holy grail of visualization systems makes exploring different data facets so intuitive, and recommends views that are so relevant, that users rapidly converge onto valuable insights – irrespective of dataset size. Unfortunately, existing systems fall far short of this goal [16].

We propose instead to blend these two systems into a *Data Visualization Management System* (DVMS) to make all database features available for visualization. A DVMS scales common data transformations to massive datasets by executing them directly in the database. Furthermore, a DVMS can leverage databases to support interactivity with little effort. For example, with lineage query support, we can automatically link related geometric objects (e.g., circles, rectangles) across views by tracking overlap in the input records that generated them. DVMS's can also incorporate novel *visual optimizations* to reduce rendering latencies, such as: (1) applying occlusion filters to remove records that render as geometric objects hidden from the user's view; (2) output-based downsampling of datasets to match the viewport size; and (3) rendering on both the client and server to balance resource and network constraints.

We present the two central design ideas behind our proposed DVMS Ermac: (1) the user specifies a visualization workflow, or mapping from raw data to geometric objects, using a *declarative visualization language*; and (2) the workflow is compiled into relational algebra queries, which are ex-

Progressive Visual Analytics: User-Driven Visual Exploration of In-Progress Analytics

Charles D. Stolper, Adam Perer, and David Gotz

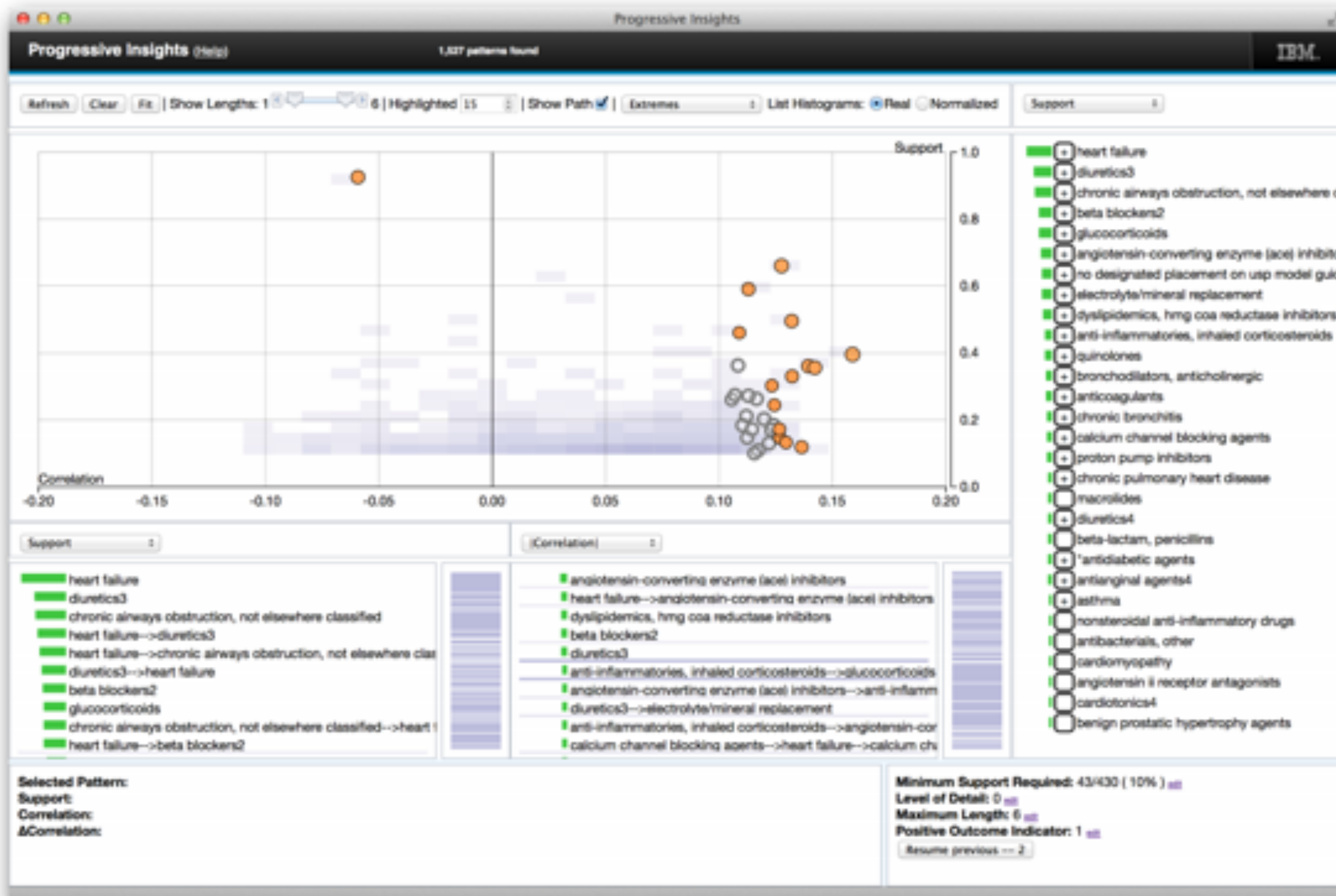


Fig. 1. The Progressive Insights system features progressive visual analytics, and supports user-driven exploration of in-progress analytics. Partial results from the progressive analytics enhance the scatterplot, list, and tree visualizations without interfering with users' cognitive workflow.

Abstract— As datasets grow and analytic algorithms become more complex, the typical workflow of analysts launching an analytic, waiting for it to complete, inspecting the results, and then re-launching the computation with adjusted parameters is not realistic for many real-world tasks. This paper presents an alternative workflow, *progressive visual analytics*, which enables an analyst to inspect partial results of an algorithm as they become available and interact with the algorithm to prioritize subspaces of interest. Progressive visual analytics depends on adapting analytical algorithms to produce meaningful partial results and enable analyst intervention without sacrificing computational speed. The paradigm also depends on adapting information visualization techniques to incorporate