

# CS 295B/CS 395B

## Systems for Knowledge Discovery

Lecture 4: Formal  
Language Design as  
Method



The University of Vermont

# FlumeJava: Easy, Efficient Data-Parallel Pipelines

Craig Chambers, Ashish Raniwala, Frances Perry,  
Stephen Adams, Robert R. Henry,  
Robert Bradshaw, Nathan Weizenbaum  
Google, Inc.  
{chambers,raniwala,fjp,sra,rrh,robertwb,nweiz}@google.com

## Abstract

MapReduce and similar systems significantly ease the task of writing data-parallel code. However, many real-world computations require a pipeline of MapReduces, and programming and managing such pipelines can be difficult. We present FlumeJava, a Java library that makes it easy to develop, test, and run efficient data-parallel pipelines. At the core of the FlumeJava library are a couple of classes that represent immutable parallel collections, each supporting a modest number of operations for processing them in parallel. Parallel collections and their operations present a simple, high-level, uniform abstraction over different data representations and execution strategies. To enable parallel operations to run efficiently, FlumeJava defers their evaluation, instead internally constructing an execution plan dataflow graph. When the final results of the parallel operations are eventually needed, FlumeJava first optimizes the execution plan, and then executes the optimized operations on appropriate underlying primitives (e.g., MapReduces). The combination of high-level abstractions for parallel data and computation, deferred evaluation and optimization, and efficient parallel primitives yields an easy-to-use system that approaches the efficiency of hand-optimized pipelines. FlumeJava is in active use by hundreds of pipeline developers within Google.

**Categories and Subject Descriptors** D.1.3 [Concurrent Programming]: Parallel Programming

**General Terms** Algorithms, Languages, Performance

**Keywords** data-parallel programming, MapReduce, Java

## 1. Introduction

Building programs to process massive amounts of data in parallel can be very hard. MapReduce [6–8] greatly eased this task for data-parallel computations. It presented a simple abstraction to users for how to think about their computation, and it managed many of the difficult low-level tasks, such as distributing and coordinating the parallel work across many machines, and coping robustly with failures of machines, networks, and data. It has been used very successfully in practice by many developers. MapReduce’s success in this domain inspired the development of a number of related systems, including Hadoop [2], LINQ/Dryad [20], and Pig [3].

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

PLDI’10, June 5–10, 2010, Toronto, Ontario, Canada  
Copyright © 2010 ACM 978-1-4503-0019-3/10/06...\$10.00

MapReduce works well for computations that can be broken down into a map step, a shuffle step, and a reduce step, but for many real-world computations, a chain of MapReduce stages is required. Such data-parallel *pipelines* require additional coordination code to chain together the separate MapReduce stages, and require additional work to manage the creation and later deletion of the intermediate results between pipeline stages. The logical computation can become obscured by all these low-level coordination details, making it difficult for new developers to understand the computation. Moreover, the division of the pipeline into particular stages becomes “baked in” to the code and difficult to change later if the logical computation needs to evolve.

In this paper we present FlumeJava, a new system that aims to support the development of data-parallel pipelines. FlumeJava is a Java library centered around a few classes that represent *parallel collections*. Parallel collections support a modest number of *parallel operations* which are composed to implement data-parallel computations. An entire pipeline, or even multiple pipelines, can be implemented in a single Java program using the FlumeJava abstractions; there is no need to break up the logical computation into separate programs for each stage.

FlumeJava’s parallel collections abstract away the details of how data is represented, including whether the data is represented as an in-memory data structure, as one or more files, or as an external storage service such as a MySQL database or a Bigtable [5]. Similarly, FlumeJava’s parallel operations abstract away their implementation strategy, such as whether an operation is implemented as a local sequential loop, or as a remote parallel MapReduce invocation, or (in the future) as a query on a database or as a streaming computation. These abstractions enable an entire pipeline to be initially developed and tested on small in-memory test data, running in a single process, and debugged using standard Java IDEs and debuggers, and then run completely unchanged over large production data. They also confer a degree of adaptability of the logical FlumeJava computations as new data storage mechanisms and execution services are developed.

To achieve good performance, FlumeJava internally implements parallel operations using *deferred evaluation*. The invocation of a parallel operation does not actually run the operation, but instead simply records the operation and its arguments in an internal *execution plan* graph structure. Once the execution plan for the whole computation has been constructed, FlumeJava *optimizes* the execution plan, for example *fusing* chains of parallel operations together into a small number of MapReduce operations. FlumeJava then runs the optimized execution plan. When running the execution plan, FlumeJava chooses which strategy to use to implement each operation (e.g., local sequential loop vs. remote parallel MapReduce, based in part on the size of the data being processed), places remote computations near the data they operate on, and per-

# Paper context

- Complement paper to Spark SQL
  - Why Spark SQL + CaRL pairing?
- DML SQL: what is it good for?
  - Ad hoc queries in SQL
  - Output of SQL query → input to s.t. else

# MapReduce: Simplified Data Processing on Large Clusters

Jeffrey Dean and Sanjay Ghemawat

jeff@google.com, sanjay@google.com

Google, Inc.

## Abstract

MapReduce is a programming model and an associated implementation for processing and generating large data sets. Users specify a *map* function that processes a key/value pair to generate a set of intermediate key/value pairs, and a *reduce* function that merges all intermediate values associated with the same intermediate key. Many real world tasks are expressible in this model, as shown in the paper.

Programs written in this functional style are automatically parallelized and executed on a large cluster of commodity machines. The run-time system takes care of the details of partitioning the input data, scheduling the program's execution across a set of machines, handling machine failures, and managing the required inter-machine communication. This allows programmers without any experience with parallel and distributed systems to easily utilize the resources of a large distributed system.

Our implementation of MapReduce runs on a large cluster of commodity machines and is highly scalable: a typical MapReduce computation processes many terabytes of data on thousands of machines. Programmers find the system easy to use: hundreds of MapReduce programs have been implemented and upwards of one thousand MapReduce jobs are executed on Google's clusters every day.

## 1 Introduction

Over the past five years, the authors and many others at Google have implemented hundreds of special-purpose computations that process large amounts of raw data, such as crawled documents, web request logs, etc., to compute various kinds of derived data, such as inverted indices, various representations of the graph structure of web documents, summaries of the number of pages crawled per host, the set of most frequent queries in a

given day, etc. Most such computations are conceptually straightforward. However, the input data is usually large and the computations have to be distributed across hundreds or thousands of machines in order to finish in a reasonable amount of time. The issues of how to parallelize the computation, distribute the data, and handle failures conspire to obscure the original simple computation with large amounts of complex code to deal with these issues.

As a reaction to this complexity, we designed a new abstraction that allows us to express the simple computations we were trying to perform but hides the messy details of parallelization, fault-tolerance, data distribution and load balancing in a library. Our abstraction is inspired by the *map* and *reduce* primitives present in Lisp and many other functional languages. We realized that most of our computations involved applying a *map* operation to each logical "record" in our input in order to compute a set of intermediate key/value pairs, and then applying a *reduce* operation to all the values that shared the same key, in order to combine the derived data appropriately. Our use of a functional model with user-specified map and reduce operations allows us to parallelize large computations easily and to use re-execution as the primary mechanism for fault tolerance.

The major contributions of this work are a simple and powerful interface that enables automatic parallelization and distribution of large-scale computations, combined with an implementation of this interface that achieves high performance on large clusters of commodity PCs.

Section 2 describes the basic programming model and gives several examples. Section 3 describes an implementation of the MapReduce interface tailored towards our cluster-based computing environment. Section 4 describes several refinements of the programming model that we have found useful. Section 5 has performance measurements of our implementation for a variety of tasks. Section 6 explores the use of MapReduce within Google including our experiences in using it as the basis

# Paper context: related work

- MapReduce (Google, OSDI 2004)

## MapReduce: Simplified Data Processing on Large Clusters



Apache Hadoop

Download

Documentation ▾ Community ▾ Development ▾ Help ▾

Apache Software Foundation



Apache Hadoop

The Apache™ Hadoop® project develops open-source software for reliable, scalable, distributed computing.

The Apache Hadoop software library is a framework that allows for the distributed processing of large data sets across clusters of computers using simple programming models. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage. Rather than rely on hardware to deliver high-availability, the library itself is designed to detect and handle failures at the application layer, so delivering a highly-available service on top of a cluster of computers, each of which may be prone to failures.

[Learn more »](#)

[Download »](#)

[Getting started »](#)

### Latest news

Release 3.3.1 available

2021 Jun 15

This is the first stable release of Apache Hadoop 3.3.x line. It contains 697 bug fixes, improvements and enhancements since 3.3.0.

Users are encouraged to read the [overview of major changes](#) since 3.3.0. For details of 697 bug fixes, improvements, and other enhancements since the previous 3.3.0 release, please check [release notes](#) and [changelog](#) detail the changes since 3.3.0.

Ozone 1.1.0 is released

2021 Apr 17

General available(GA) release of Apache Hadoop Ozone with Volume/Bucket Quota Support, Security related enhancements, ofs/o3fs performance improvements, Recon improvements etc.

For more information check the [ozone site](#).

### Modules

The project includes these modules:

- **Hadoop Common:** The common utilities that support the other Hadoop modules.
- **Hadoop Distributed File System (HDFS™):** A distributed file system that provides high-throughput access to application data.
- **Hadoop YARN:** A framework for job scheduling and cluster resource management.
- **Hadoop MapReduce:** A YARN-based system for parallel processing of large data sets.

### Who Uses Hadoop?

A wide variety of companies and organizations use Hadoop for both research and production. Users are encouraged to add themselves to the Hadoop PoweredBy [wiki page](#).

### Related projects

Other Hadoop-related projects at Apache include:

- **Ambari™:** A web-based tool for provisioning, managing, and monitoring Apache Hadoop clusters which includes support for Hadoop HDFS, Hadoop MapReduce, Hive, HCatalog, HBase, ZooKeeper, Oozie, Pig and Sqoop. Ambari also provides a dashboard for viewing cluster health such as heatmaps and ability to view MapReduce, Pig and Hive applications visually alongwith features to diagnose their performance characteristics in a user-friendly manner.
- **Avro™:** A data serialization system.
- **Cassandra™:** A scalable multi-master database with no single points of failure.
- **Chukwa™:** A data collection system for managing large distributed systems.
- **HBase™:** A scalable, distributed database that supports structured data storage for large tables.
- **Hive™:** A data warehouse infrastructure that provides data summarization and ad hoc querying.
- **Mahout™:** A Scalable machine learning and data mining library

# Paper context: related work

- MapReduce (Google, OSDI 2004)
- Hadoop (Yahoo! → Apache, 2006)



## Pig Latin: A Not-So-Foreign Language for Data Processing

Christopher Olston<sup>\*</sup>  
Yahoo! Research

Benjamin Reed<sup>†</sup>  
Yahoo! Research

Utkarsh Srivastava<sup>‡</sup>  
Yahoo! Research

Ravi Kumar<sup>§</sup>  
Yahoo! Research

Andrew Tomkins<sup>¶</sup>  
Yahoo! Research

### ABSTRACT

There is a growing need for ad-hoc analysis of extremely large data sets, especially at internet companies where innovation critically depends on being able to analyze terabytes of data collected every day. Parallel database products, e.g., Teradata, offer a solution, but are usually prohibitively expensive at this scale. Besides, many of the people who analyze this data are entrenched procedural programmers, who find the declarative, SQL style to be unnatural. The success of the more procedural *map-reduce* programming model, and its associated scalable implementations on commodity hardware, is evidence of the above. However, the *map-reduce* paradigm is too low-level and rigid, and leads to a great deal of custom user code that is hard to maintain, and reuse.

We describe a new language called *Pig Latin* that we have designed to fit in a sweet spot between the declarative style of SQL, and the low-level, procedural style of map-reduce. The accompanying system, Pig, is fully implemented, and compiles Pig Latin into physical plans that are executed over *Hadoop*, an open-source, map-reduce implementation. We give a few examples of how engineers at Yahoo! are using Pig to dramatically reduce the time required for the development and execution of their data analysis tasks, compared to using Hadoop directly. We also report on a novel debugging environment that comes integrated with Pig, that can lead to even higher productivity gains. Pig is an open-source, Apache-incubator project, and available for general use.

### Categories and Subject Descriptors:

H.2.3 Database Management: Languages

General Terms: Languages

<sup>\*</sup>olston@yahoo-inc.com

<sup>†</sup>breed@yahoo-inc.com

<sup>‡</sup>utkarsh@yahoo-inc.com

<sup>§</sup>ravikuma@yahoo-inc.com

<sup>¶</sup>atomkins@yahoo-inc.com

### 1. INTRODUCTION

At a growing number of organizations, innovation revolves around the collection and analysis of enormous data sets such as web crawls, search logs, and click streams. Internet companies such as Amazon, Google, Microsoft, and Yahoo! are prime examples. Analysis of this data constitutes the innermost loop of the product improvement cycle. For example, the engineers who develop search engine ranking algorithms spend much of their time analyzing search logs looking for exploitable trends.

The sheer size of these data sets dictates that it be stored and processed on highly parallel systems, such as shared-nothing clusters. Parallel database products, e.g., Teradata, Oracle RAC, Netezza, offer a solution by providing a simple SQL query interface and hiding the complexity of the physical cluster. These products however, can be prohibitively expensive at web scale. Besides, they wrench programmers away from their preferred method of analyzing data, namely writing imperative scripts or code, toward writing declarative queries in SQL, which they often find unnatural, and overly restrictive.

As evidence of the above, programmers have been flocking to the more procedural *map-reduce* [4] programming model. A *map-reduce* program essentially performs a group-by-aggregation in parallel over a cluster of machines. The programmer provides a map function that dictates how the grouping is performed, and a reduce function that performs the aggregation. What is appealing to programmers about this model is that there are only two high-level declarative primitives (*map* and *reduce*) to enable parallel processing, but the rest of the code, i.e., the *map* and *reduce* functions, can be written in any programming language of choice, and without worrying about parallelism.

Unfortunately, the *map-reduce* model has its own set of limitations. Its one-input, two-stage data flow is extremely rigid. To perform tasks having a different data flow, e.g., joins or  $n$  stages, inelegant workarounds have to be devised. Also, custom code has to be written for even the most common operations, e.g., projection and filtering. These factors

# Paper context: related work

- MapReduce (Google, OSDI 2004)
- Hadoop (Yahoo! → Apache, 2006)
- Pig Latin (Yahoo!, SIGMOD 2008)



The A

The A  
of con  
offering  
detect  
of whic

Learn

Latest

Rel

This  
Had  
imp  
3.3.Use  
of n  
697  
enh  
rele  
cha

Ozc

Ger  
Hac  
Sup  
ofs/  
imp  
For

## Pig Latin: A Not-So-Foreign Language for Data Processing

### Hive – A Petabyte Scale Data Warehouse Using Hadoop

Ashish Thusoo, Joydeep Sen Sarma, Namit Jain, Zheng Shao, Prasad Chakka, Ning Zhang, Suresh Antony, Hao Liu and Raghavam Murthy

*Facebook Data Infrastructure Team*

**Abstract**— The size of data sets being collected and analyzed in the industry for business intelligence is growing rapidly, making traditional warehousing solutions prohibitively expensive. Hadoop [1] is a popular open-source map-reduce implementation which is being used in companies like Yahoo, Facebook etc. to store and process extremely large data sets on commodity hardware. However, the map-reduce programming model is very low level and requires developers to write custom programs which are hard to maintain and reuse. In this paper, we present *Hive*, an open-source data warehousing solution built on top of Hadoop. Hive supports queries expressed in a SQL-like declarative language - *HiveQL*, which are compiled into map-reduce jobs that are executed using Hadoop. In addition, *HiveQL* enables users to plug in custom map-reduce scripts into queries. The language includes a type system with support for tables containing primitive types, collections like arrays and maps, and nested compositions of the same. The underlying IO libraries can be extended to query data in custom formats. Hive also includes a system catalog - *Metastore* - that contains schemas and statistics, which are useful in data exploration, query optimization and query compilation. In Facebook, the Hive warehouse contains tens of thousands of tables and stores over 700TB of data and is being used extensively for both reporting and ad-hoc analyses by more than 200 users per month.

#### I. INTRODUCTION

Scalable analysis on large data sets has been core to the functions of a number of teams at Facebook - both engineering and non-engineering. Apart from ad hoc analysis and business intelligence applications used by analysts across the company, a number of Facebook products are also based on analytics. These products range from simple reporting and

data. As a result we started exploring Hadoop as a technology to address our scaling needs. The fact that Hadoop was already an open source project that was being used at petabyte scale and provided scalability using commodity hardware was a very compelling proposition for us. The same jobs that had taken more than a day to complete could now be completed within a few hours using Hadoop.

However, using Hadoop was not easy for end users, especially for those users who were not familiar with map-reduce. End users had to write map-reduce programs for simple tasks like getting raw counts or averages. Hadoop lacked the expressiveness of popular query languages like SQL and as a result users ended up spending hours (if not days) to write programs for even simple analysis. It was very clear to us that in order to really empower the company to analyze this data more productively, we had to improve the query capabilities of Hadoop. Bringing this data closer to users is what inspired us to build Hive in January 2007. Our vision was to bring the familiar concepts of tables, columns, partitions and a subset of SQL to the unstructured world of Hadoop, while still maintaining the extensibility and flexibility that Hadoop enjoyed. Hive was open sourced in August 2008 and since then has been used and explored by a number of Hadoop users for their data processing needs.

Right from the start, Hive was very popular with all users within Facebook. Today, we regularly run thousands of jobs on the Hadoop/Hive cluster with hundreds of users for a wide variety of applications starting from simple summarization jobs to business intelligence, machine learning applications and to also support Facebook product features.

# Paper context: related work

- MapReduce (Google, OSDI 2004)
- Hadoop (Yahoo! → Apache, 2006)
- Pig Latin (Yahoo!, SIGMOD 2008)
- Hive (Facebook, IEEE ICDE 2010)

The screenshot shows the Apache Hadoop homepage with the title "MapReduce: Simplified Data Processing on Large Clusters" at the top. Below it, there's a section for "Pig Latin: A Not-So-Foreign Language for Data Processing", "Hive – A Petabyte Scale Data Warehouse Using", and "FlumeJava: Easy, Efficient Data-Parallel Pipelines". The sidebar on the left contains links like "Latest", "Rel...", "This Had...", "Use of n...", "enh...", "rele...", "cha...", "Oz...", "Ger...", "Hac...", "Sup...", "ots/...", "imp...", and "For...".

# Paper context: related work

- MapReduce (Google, OSDI 2004)
- Hadoop (Yahoo! → Apache, 2006)
- Pig Latin (Yahoo!, SIGMOD 2008)
- Hive (Facebook, IEEE ICDE 2010)
- FlumeJava (Google, PLDI 2010)
- ...
- Spark SQL (Academia, SIGMOD 2015)

The screenshot shows the Apache Hadoop homepage with a large, stylized blue star in the center. The star has several points, each containing a different framework name: "Pig Latin: A Not-So-Foreign Language", "Hive – A Column-oriented Database", "FlumeJava", and "LINQ (MSR, SIGMOD 2006)". The background of the star is a light gray gradient.

**Pig Latin: A Not-So-Foreign Language**

**Hive – A Column-oriented Database**

**FlumeJava**

**LINQ (MSR, SIGMOD 2006)**

**Abstract**

MapReduce and similar systems significantly ease the task of writing data-parallel code. However, many real-world computations require a pipeline of MapReduces, and programming and managing such pipelines can be difficult. We present FlumeJava, a Java library that makes it easy to develop, test, and run efficient data-parallel pipelines. At the core of the FlumeJava library are a couple of classes that represent immutable parallel collections, each supporting a modest number of operations for processing them in parallel. Parallel collections and their operations present a simple, high-level, uniform abstraction over different data representations and execution strategies. To enable parallel operations to run efficiently, FlumeJava defers their evaluation, instead internally constructing an execution plan dataflow graph. When the final results of the parallel operations are eventually needed, FlumeJava first optimizes the execution plan, and then executes the optimized operations on appropriate underlying primitives (e.g., MapReduces). The combination of high-level abstractions for parallel data and computation, deferred evaluation and optimization, and efficient parallel execution make FlumeJava a powerful and flexible system for building data-parallel pipelines.

**MapReduce works well for computations that can be broken down into a map step, a shuffle step, and a reduce step, but for many real-world computations, a chain of MapReduce stages is required. Such data-parallel *pipelines* require additional coordination code to chain together the separate MapReduce stages, and require additional work to manage the creation and later deletion of the intermediate results between pipeline stages. The logical computation can become obscured by all these low-level coordination details, making it difficult for new developers to understand the computation. Moreover, the division of the pipeline into particular stages becomes “baked in” to the code and difficult to change later if the logical computation needs to evolve.**

In this paper we present FlumeJava, a new system that aims to support the development of data-parallel pipelines. FlumeJava is a Java library centered around a few classes that represent *parallel collections*. Parallel collections support a modest number of *parallel operations* which are composed to implement data-parallel computations. An entire pipeline, or even multiple pipelines, can be implemented in a single Java program using the FlumeJava ab-

# Paper context: related work

- MapReduce (Google, OSDI 2004)

- Hadoop (Yahoo! → Apache, 2006)

- Pig Latin (Yahoo!, SIGMOD 2008)

- Hive (Facebook, IEEE ICDE 2010)

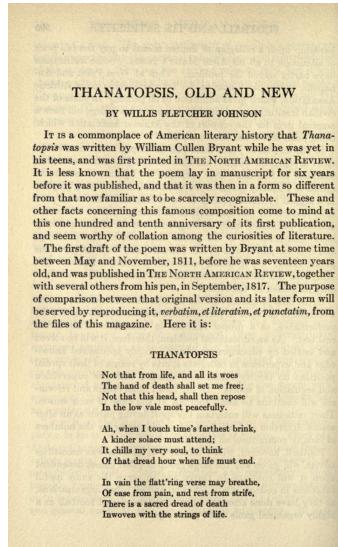
- FlumeJava (Google, PLDI 2010)

- ...

- Spark SQL (Academia, SIGMOD 2015)

# Paper context: data format

Raw non-digital text



Webpage source

```
<!DOCTYPE html>
<html>
  <head>
    <title>Virtual Teaching Doesn't Mean Giving Up on the Live Lecture</title>
    <link rel='preconnect' href='https://fonts.gstatic.com'>
    <link href='https://fonts.googleapis.com/css2?family=Crimson+Text:wght@400;600&display=swap' rel='stylesheet'>
    <link rel='stylesheet' type='text/css' href='style.css'>
  </head>
  <body>
    <div class='home_container'>
      <div class='paper_title'>Virtual Teaching Doesn't Mean Giving Up on the Live Lecture</div>
      <div>
        <a href='http://graphics.stanford.edu/~kayonf/'>Keyvan Fatahalian</a>
      </div>
      <div style='border-bottom: 1px solid #C0C0C0; padding-bottom: 10px;'>Jan 13, 2021</div>
      <div>
        <img src='images/teaser.jpg' width='720' height='490' class='inline_image' alt='teaser image of classroom main page'>
      </div>
      <p>Last quarter was, surprisingly, one of my most enjoyable quarters teaching. Lecture attendance was up, student-staff interaction was up, and our co-instructor, Dr. Kunal Olukotun, and I felt we had a better handle than ever before on whether students were following the lecture. By the end of the quarter, a higher percentage of the class was attending live lectures than previous quarters. "The main difference between this last quarter and my previous nine years of teaching? This 100+ student course took place in a few virtual classrooms that I designed. I first taught on Zoom in Spring 2020 and got a sense of what worked and didn't work virtually. When it came time to teach "Parallel Computing" in Fall 2020, I decided to pitch my co-instructor Kunal on a new, experimental format. " After using my virtual classroom, I don't want to go back to vanilla in-person teaching. The virtual classroom has made it easy for me to experiment with different enhanced teaching mechanisms I'd been hearing about for years. Not only has it been easier to communicate with my students than ever before, I've also been interacting with a wider, more diverse set of students. " To help other instructors during the pandemic and beyond, I've put together this blog post explaining how I translated the main elements of an in-person classroom to video-and what enhancements I added. Note that all
      </p>
    </div>
  </body>
</html>
```

Text Formatting

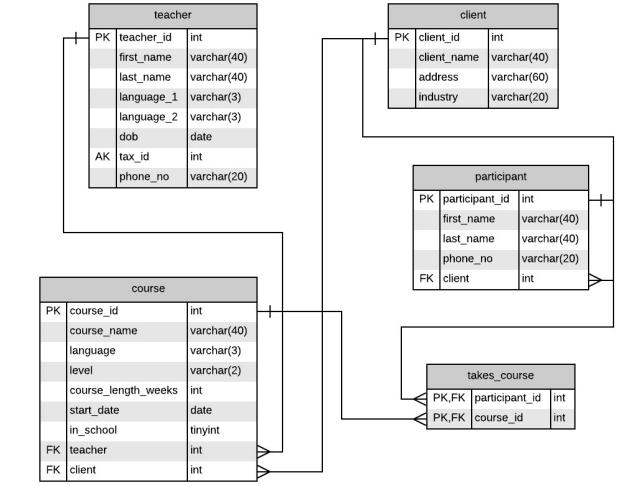
Nested markup tags

Structure

Pandas Dataframe

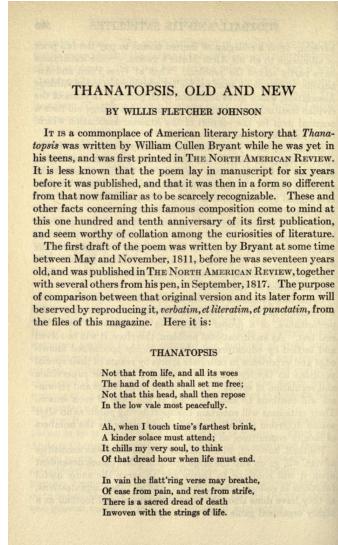
In [10]:	df.iloc[3400]	
Out[10]:	agent_name seed t action missed_ball xpos_ball ypos_ball xpos_ball_prev ypos_ball_prev xpos_pad ypos_pad xpos_pad_prev ypos_pad_prev board_alive is_far_left is_far_right score red_width ball_speed ball_down xdist_ball_pad ydist_ball_pad 12_ball_pad num_bricks_left bricks_in_col_00 bricks_in_col_01 bricks_in_col_02 bricks_in_col_03 bricks_in_col_04 bricks_in_col_05 bricks_in_col_06 bricks_in_col_07	Target 127525 1585 right False 160.862 48.1418 159.235 51.76 160 143 160 143 False False 270 small fast False 0.862097 94.8582 94.8621 54 2 0 2 3 3 1 1 4
	60877497971844372295859610308414	

Relational Database



# Paper context: data format

Raw non-digital text



Webpage source

```
<!DOCTYPE html>
<html>
  <head>
    <title>Virtual Teaching Doesn't Mean Giving Up on the Live Lecture</title>
    <link href="preconnect" href="https://fonts.gstatic.com">
    <link href="https://fonts.googleapis.com/css2?family=Crimson+Text:wght@400;600&display=swap" rel="stylesheet">
    <link rel="stylesheet" type="text/css" href="style.css" />
  </head>
  <body>
    <div class="home_container">
      <div class="paper_title">Virtual Teaching Doesn't Mean Giving Up on the Live Lecture</div>
      <div>
        <a href="http://graphics.stanford.edu/~kayonff/Kayahanlian/>
        Stanford University</a>
      </div>
      <div style="border-bottom: 3px solid #C0C0C0; padding-bottom: 10px;">Jan 13, 2021</div>
      <div>
        
      </div>
      <p>
        Last quarter was, surprisingly, one of my most enjoyable quarters teaching. Lecture attendance was up, student-staff interaction was up, and our co-instructor, Dr. Kunal Olukotun, and I felt we had a better handle than ever before on whether students were following the lecture. By the end of the quarter, a higher percentage of the class was attending live lectures than previous quarters.
      </p>
      <p>
        The main difference between this last quarter and my previous nine years of teaching? This 100+ student course took place in a few virtual classrooms that I designed. I first taught on Zoom in Spring 2020 and got a sense of what worked and didn't work virtually. When it came time to teach in Fall 2020, I decided to pitch my co-instructor Kunal on a new, experimental format.
      </p>
      <p>
        After using my virtual classroom, I don't want to go back to vanilla in-person teaching. The virtual classroom has made it easy for me to experiment with different enhanced teaching mechanisms I'd been hearing about for years. Not only has it been easier to communicate with my students than ever before, I've also been interacting with a wider, more diverse set of students.
      </p>
      <p>
        To help other instructors during the pandemic and beyond, I've put together this blog post explaining how I translated the main elements of an in-person classroom to video-and what enhancements I added. Note that all
      </p>
    </div>
  </body>
</html>
```

Text Formatting

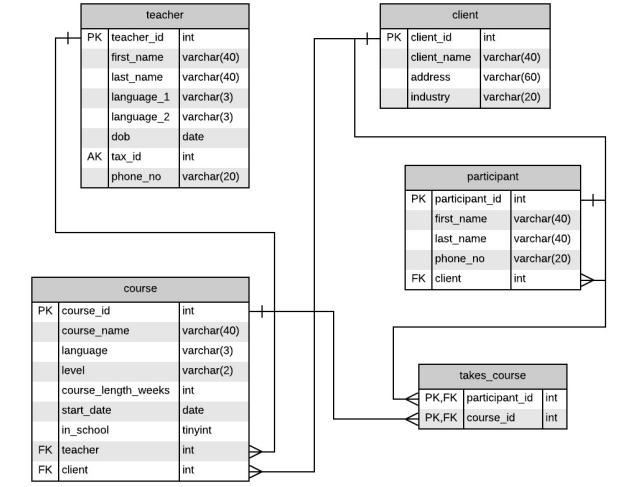
Nested markup tags

Pre-Processing

Pandas Dataframe

In [10]:	df.iloc[3400]
Out[10]:	agent_name seed t action missed_ball xpos_ball ypos_ball xpos_ball_prev ypos_ball_prev xpos_pad ypos_pad xpos_pad_prev ypos_pad_prev board_alive is_far_left is_far_right score red_width ball_speed ball_down xdist_ball_pad ydist_ball_pad 12_ball_pad num_bricks_left bricks_in_col_00 bricks_in_col_01 bricks_in_col_02 bricks_in_col_03 bricks_in_col_04 bricks_in_col_05 bricks_in_col_06 bricks_in_col_07
	Target 127525 1585 right False 160.862 48.1418 59.235 51.76 160 143 160 143 False False 270 small fast False 0.862097 94.8582 94.8621 54 2 0 2 3 3 1 1 4
	60877497971844372295859610308414

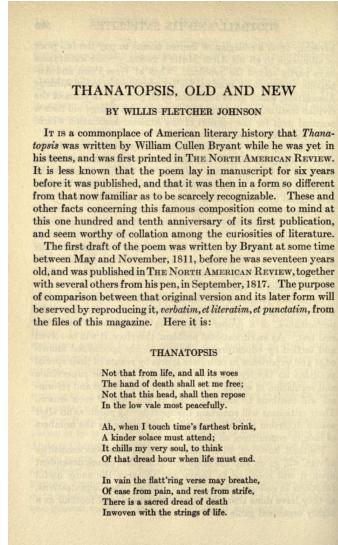
Relational Database



Typed records/objects + Relations

# Paper context: data format

Raw non-digital text



Webpage source

```
<!DOCTYPE html>
<html>
  <head>
    <title>Virtual Teaching Doesn't Mean Giving Up on the Live Lecture</title>
    <link href="preconnect" href="https://fonts.gstatic.com">
    <link href="https://fonts.googleapis.com/css2?family=Crimson+Text:wght@400;600&display=swap" rel="stylesheet">
    <link rel="stylesheet" type="text/css" href="style.css">
  </head>
  <body>
    <div class="home_container">
      <div class="paper_title">Virtual Teaching Doesn't Mean Giving Up on the Live Lecture</div>
      <div>
        <a href="http://graphics.stanford.edu/~kayonff">Keyvan Fatahalian</a>
        <br/>
        Stanford University
      </div>
      <div style="border-bottom: 1px solid #c0c0c0; padding-bottom: 10px;">Jan 13, 2021</div>
      <div>
        
      </div>
      <p>Last quarter was, surprisingly, one of my most enjoyable quarters teaching. Lecture attendance was up, student-staff interaction was up, and our co-instructor, Dr. Kunal Kulkarni (https://orcsweb.itsrcf.stanford.edu/~kukule) Olukotun, " and I felt we had a better handle than ever before on whether students were following the lecture. By the end of the quarter, a higher percentage of the class was attending live lectures than previous quarters. "</p>
      <p>The main difference between this last quarter and my previous nine years of teaching? This 100+ student course took place in a few virtual classrooms that I designed. I first taught on Zoom in Spring 2020 and got a sense of what worked and didn't work virtually. When it came time to teach in Fall 2020, I decided to pitch my co-instructor Kunal on a new, experimental format. "</p>
      <p>After using my virtual classroom, I don't want to go back to vanilla in-person teaching. The virtual classroom has made it easy for me to experiment with different enhanced teaching mechanisms I'd been hearing about for years. Not only has it been easier to communicate with my students than ever before, I've also been interacting with a wider, more diverse set of students. "</p>
      <p>To help other instructors during the pandemic and beyond, I've put together this blog post explaining how I translated the main elements of an in-person classroom to video-and what enhancements I added. Note that all
    </div>
  </body>
</html>
```

Text Formatting

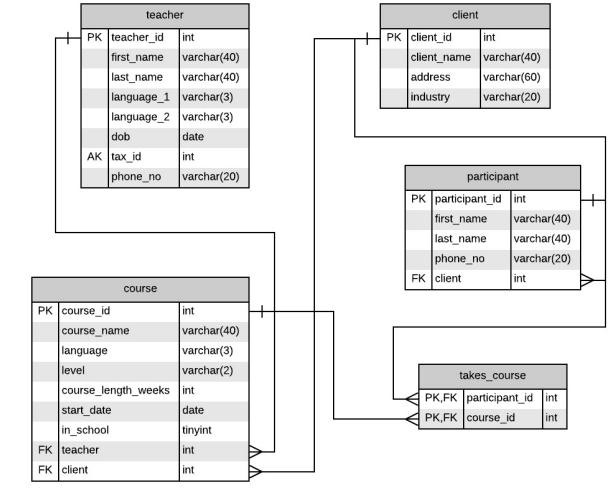
Nested markup tags

Typed records/objects  
Domain knowledge

Pandas Dataframe

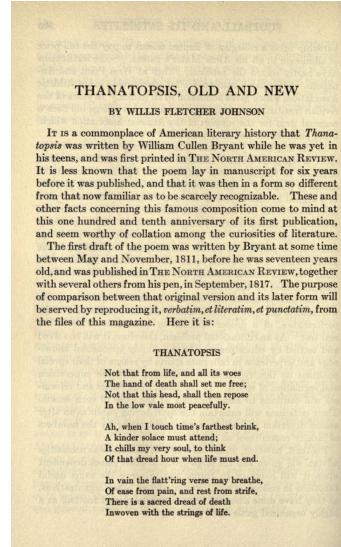
In [10]:	df.iloc[3400]	
Out[10]:	agent_name seed t action missed_ball xpos_ball ypos_ball xpos_ball_prev ypos_ball_prev xpos_pad ypos_pad xpos_pad_prev ypos_pad_prev board_alive is_far_left is_far_right score red_width ball_speed ball_down xdist_ball_pad ydist_ball_pad 12_ball_pad num_bricks_left bricks_in_col_00 bricks_in_col_01 bricks_in_col_02 bricks_in_col_03 bricks_in_col_04 bricks_in_col_05 bricks_in_col_06 bricks_in_col_07	Target 127525 1585 right False 160.862 48.1418 159.235 51.748 160 143 160 143 False False 270 small fast False 0.862097 94.8582 94.8621 54 2 0 2 3 3 1 1 4
	60877497971844372295859610308414	

Relational Database



Discussion: How might structure encode domain knowledge?

# Paper context: data format



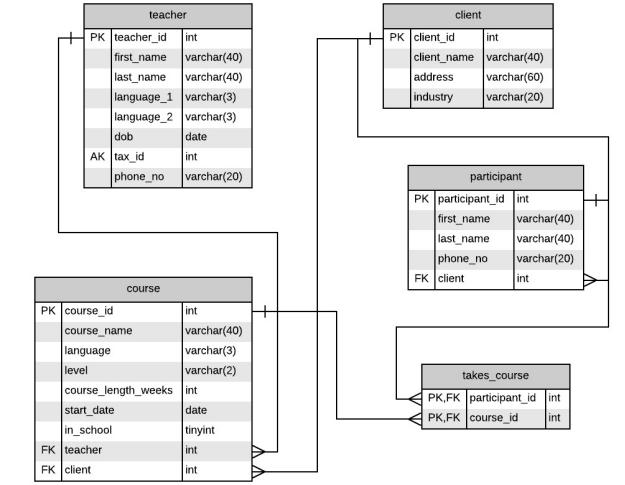
Text Formatting

Nested markup tags

```
<!DOCTYPE html>
<html>
  <head>
    <title>Virtual Teaching Doesn't Mean Giving Up on the Live Lecture</title>
    <link rel="preconnect" href="https://fonts.gstatic.com">
    <link href="https://fonts.googleapis.com/css2?family=Crimson+Text:wght@400;600&display=swap" rel="stylesheet">
    <link rel="stylesheet" type="text/css" href="style.css">
  </head>
  <body>
    <div class="home_container">
      <div class="paper_title">Virtual Teaching Doesn't Mean Giving Up on the Live Lecture</div>
      <p class="paper_authors"></p>
      <div>
        <a href="http://graphics.stanford.edu/~kayvonf/Fataliani.html">Keyvon Fataliani</a>
      </div>
      <div style="border-bottom: 1px solid #C0C0C0; padding-bottom: 10px;">Jan 13, 2021</div>
      <div>
        
      </div>
      <p>Last quarter was, surprisingly, one of my most enjoyable quarters teaching. Lecture attendance was up, student-started questions were up, and our co-instructor, Kunle Olukotun, and I felt we had a better handle than ever before on whether students were following the lecture. By the end of the quarter, a higher percentage of the class was attending live lectures than previous quarters. “</p>
      <p>“The main difference between this last quarter and my previous nine years of teaching? This 100+ student course took place in a few virtual classrooms that I designed. I first taught on Zoom in Spring 2020 and got a sense of what worked and didn’t work virtually. When it came time to teach in Fall 2020, I decided to pitch my co-instructor Kunle on a new, experimental format. “</p>
      <p>“After using my virtual classroom, I don’t want to go back to vanilla in-person teaching. The virtual classroom has made it easy for me to experiment with different enhanced teaching mechanisms I’d been hearing about for years. Not only has it been easier to communicate with my students than ever before, I’ve also been interacting with a wider, more diverse set of students. “</p>
      <p>“To help other instructors during the pandemic and beyond, I’ve put together this blog post explaining how I translated the main elements of an in-person classroom to video—and what enhancements I added. Note that all</p>
```

In [10]: df.iloc[3400]	
Out[10]:	agent_name
	seed
	t
	action
	missed_ball
	xpos_ball
	ypos_ball
	xpos_ball_prev
	ypos_ball_prev
	board_pad
	ypos_pad
	xroot_pad_prev
	ypos_pad_prev
	board_alive
	is_far_left
	is_far_right
	score
	ball_width
	ball_speed
	ball_down
	xdist_ball_pad
	ydist_ball_pad
	12_ball_pad
	num_bricks_left
	bricks_in_col_00
	bricks_in_col_01
	bricks_in_col_02
	bricks_in_col_03
	bricks_in_col_04
	bricks_in_col_05
	bricks_in_col_06
	bricks_in_col_07
Target	
127525	1585
	right
	False
160.862	48.1418
	159.235
51.748	51.748
	160
	143
	160
	143
60877497971844372295859610308414	
	False
	False
	270
	small
	fast
	False
0.862097	
94.8582	
94.8621	
54	
	2
	0
	2
	3
	3
	1
	1
	4

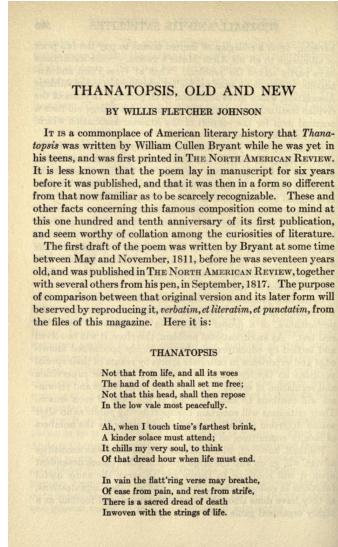
Typed records/objects



Typed records/objects + Relations

# Paper context: data format

## Text Formatting



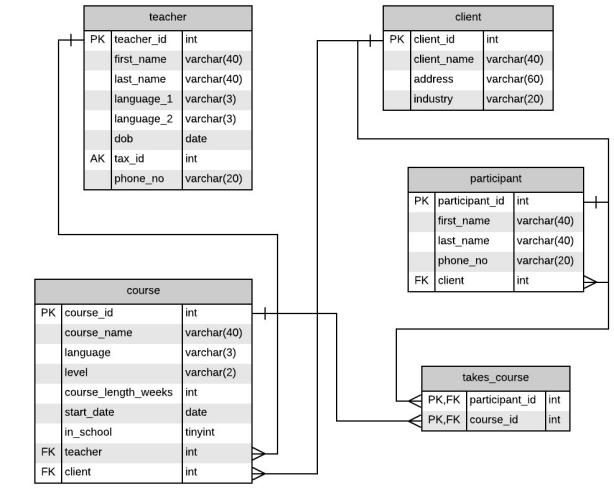
## Nested markup tags

```
<!DOCTYPE html>
<html>
  <head>
    <title>Virtual Teaching Doesn't Mean Giving Up on the Live Lecture</title>
    <link rel="preconnect" href="https://fonts.gstatic.com">
    <link href="https://fonts.googleapis.com/css2?family=Crimson+Text:wght@400;600&display=swap" rel="stylesheet">
    <link rel="stylesheet" type="text/css" href="style.css">
  </head>
  <body>
    <div class="home_container">
      <div class="paper_title">Virtual Teaching Doesn't Mean Giving Up on the Live Lecture</div>
      <div class="paper_authors"></div>
      <div href="http://graphics.stanford.edu/~kayvonf/Fataliani/>
        <div>
          <div style="border-bottom: 1px solid #C0C0C0; padding-bottom: 10px;">Jan 13, 2021</div>
          
        </div>
        <p>Last quarter was, surprisingly, one of my most enjoyable quarters teaching. Lecture attendance was up, student-staff interaction was up, and our co-instructor, Kunle Olukotun, " and I felt we had a better handle than ever before on whether students were following the lecture. By the end of the quarter, a higher percentage of the class was attending live lectures than previous quarters. "</p>
        <p>The main difference between this last quarter and my previous nine years of teaching? This 100+ student course took place in a few virtual classrooms that I designed. I first taught on Zoom in Spring 2020 and got a sense of what worked and didn't work virtually. When it came time to teach in Fall 2020, I decided to pitch my co-instructor Kunle on a new, experimental format. "</p>
        <p>After using my virtual classroom, I don't want to go back to vanilla in-person teaching. The virtual classroom has made it easy for me to experiment with different enhanced teaching mechanisms I'd been hearing about for years. Not only has it been easier to communicate with my students than ever before, I've also been interacting with a wider, more diverse set of students. "</p>
        <p>To help other instructors during the pandemic and beyond, I've put together this blog post explaining how I translated the main elements of an in-person classroom to video-and what enhancements I added. Note that all</p>
    </div>
  </body>
</html>
```

## Typed records/objects

In [10]:	df.iloc[3400]
Out[10]:	agent_name seed t action missed_ball xpos_ball ypos_ball xpos_ball_prev ypos_ball_prev xpos_pad ypos_pad xroot_pad_prev yroot_pad_prev board_alive is_far_left is_far_right score red_width ball_speed ball_down xdist_ball_pad ydist_ball_pad 12_ball_pad num_bricks_left bricks_in_col_00 bricks_in_col_01 bricks_in_col_02 bricks_in_col_03 bricks_in_col_04 bricks_in_col_05 bricks_in_col_06 bricks_in_col_07
	Target 127525 1585 right False 160.862 48.1418 159.235 51.76 160 143 160 143 False False 270 small fast False 0.862097 94.8582 94.8621 54 2 0 2 3 3 1 1 4
	60877497971844372295859610308414

## Typed records/objects + Relations



## MapReduce

Indri

## FlumeJava

## Hive

## Spark

## Indri: A language-model based search engine for complex queries (extended version)

Trevor Strohman, Donald Metzler, Howard Turtle and W. Bruce Croft

Center for Intelligence Information Retrieval  
University of Massachusetts Amherst  
Amherst, MA, 01003, USA  
[strohman@cs.umass.edu](mailto:strohman@cs.umass.edu)

**Keywords:** Search and Retrieval, Question Answering

### Abstract

Search engines are a critical tool for intelligence analysis. A number of innovations for search have been introduced since research with an emphasis on analyst needs began in the TIPSTER project. For example, the Inquiry search engine introduced support for specification of complex queries in a probabilistic inference network framework. Recent research on language modeling has led to the development of Indri, a search engine that combines the best features of inference nets and language modeling in an architecture designed for large-scale applications. In this paper, we describe the Indri system and show how the query language is designed to support modern language technologies. We also present results demonstrating that Indri is both effective and efficient.

### 1. Introduction

Search and detection technology has been a focus of DARPA and ARDA research programs since the TIPSTER program began in the early 1990s (Harman 1992). A number of innovations have been developed in this research, resulting in very significant improvements in the effectiveness of search tools. The Inquiry search engine (Callan *et al.* 1995), developed at the University of Massachusetts for the TIPSTER project, provided a query language capable of representing complex queries in a probabilistic framework and was used in a number of government and commercial applications.

In the years since Inquiry was developed, there has been significant progress, both in terms of information retrieval (IR) research and in the development of other language technologies and applications, such as information extraction and question answering. These new technologies interact with search and provide new requirements for a search engine. In addition, the ever-increasing volume of searchable data requires that search engines be scalable to the level

of multi-terabytes. In response to these requirements, we have recently developed Indri, a scalable search engine that combines the advantages of the inference net framework used in Inquiry with the language modeling approach to retrieval that has been the subject of much recent IR research (Croft and Lafferty 2003). Indri is part of the ARDA-sponsored Lemur project<sup>1</sup>.

The Indri search engine is designed to address the following goals:

- The query language should support complex queries involving evidence combination and the ability to specify a wide variety of constraints involving proximity, syntax, extracted entities, and document structure.
- The retrieval model should provide superior effectiveness across a range of query and document types (e.g. Web, cross-lingual, ad-hoc<sup>2</sup>).
- The query language and retrieval model should support retrieval at different levels of granularity (e.g. sentence, passage, XML field, document, multi-document).
- The system architecture should support very large databases, multiple databases, optimized query execution, fast indexing, concurrent indexing and querying, and portability.

In this paper, we describe the most important aspects of the Indri retrieval model, query language, and system architecture. We give some examples of the types of complex queries that can be supported, and illustrate the effectiveness and efficiency of the system using results from the 2004 TREC Terabyte track.

<sup>1</sup> <http://www.lemurproject.org>. Indri is available as a download from this site.

<sup>2</sup> “ad-hoc” refers to the TREC track that focuses on finding as many relevant documents as possible using queries of varying complexity

# Paper context

- Venue – not top-tier
- Authors: big names in IR
  - Have many other, better papers
  - 650 citations – not bad!
- Writing not most accessible
  - Very short
  - Highly specialized target audience

**So why I did I choose this paper?**

# Few papers document the intermediate DSL

<b>Operator</b>	<b>Name</b>	<b>Description</b>
#uwN(t <sub>1</sub> t <sub>2</sub> ...)	Unordered Window	Matches unordered text
#odN(t <sub>1</sub> t <sub>2</sub> ...)	Ordered Window	Matches ordered text
#any: <i>field</i>	Any operator	Finds any text appearing in a field named <i>field</i>
term. <i>field</i>	Field restriction	Finds the word <i>term</i> appearing in a field named <i>field</i>
#combine(q <sub>1</sub> q <sub>2</sub> ... )	Combine operator	Combines beliefs from other operators to form a single score for a document
#weight( w <sub>1</sub> q <sub>1</sub> w <sub>2</sub> q <sub>2</sub> ... )	Weight operator	Combines beliefs from other operators to form a single score for a document, using weights to indicate which operators should be trusted most
#greater( <i>field</i> n )	Numeric range operators	Finds any occurrence of <i>field</i> with a numeric value less than, greater than, or equal to <i>n</i>
#less( <i>field</i> n )		
#equal( <i>field</i> n )	Date range operators	
#date:before( d )		Finds any occurrence of a date occurring before or after a date, or between two dates.
#date:after( d )		
#date:between( b a )		
#operator[ <i>field</i> ](q <sub>1</sub> q <sub>2</sub> ... )	Extent retrieval	Evaluates <i>operator</i> on every occurrence of <i>field</i> ; useful for passage retrieval
#filrej( c s )	Filter reject	Evaluate the expression <i>s</i> only if <i>c</i> is not satisfied
#filreq( c s )	Filter require	Evaluate the expression <i>s</i> only if <i>c</i> is satisfied

Table 1: Indri query language operators

Consider the following information need: “I want paragraphs from news feed articles published between 1991 and 2000 that mention a person, a monetary amount, and the company InfoCom.”

This need can be expressed in the following Indri query:

```
#filreq(
  #band( NewsFeeddoctype
    #date:between(1991 2000) )
  #combine[paragraph](
    #any:person
    #any:money InfoCom ) )
```

# How does this differ from SQL?

Operator	Name	Description
#uwN(t <sub>1</sub> t <sub>2</sub> ...)	Unordered Window	Matches unordered text
#odN(t <sub>1</sub> t <sub>2</sub> ...)	Ordered Window	Matches ordered text
#any: <i>field</i>	Any operator	Finds any text appearing in a field named <i>field</i>
term. <i>field</i>	Field restriction	Finds the word <i>term</i> appearing in a field named <i>field</i>
#combine(q <sub>1</sub> q <sub>2</sub> ... )	Combine operator	Combines beliefs from other operators to form a single score for a document
#weight( w <sub>1</sub> q <sub>1</sub> w <sub>2</sub> q <sub>2</sub> ... )	Weight operator	Combines beliefs from other operators to form a single score for a document, using weights to indicate which operators should be trusted most
#greater( <i>field</i> n )	Numeric range operators	Finds any occurrence of <i>field</i> with a numeric value less than, greater than, or equal to <i>n</i>
#less( <i>field</i> n )		
#equal( <i>field</i> n )		
#date:before( d )	Date range operators	Finds any occurrence of a date occurring before or after a date, or between two dates.
#date:after( d )		
#date:between( b a )		
#operator[ <i>field</i> ](q <sub>1</sub> q <sub>2</sub> ... )	Extent retrieval	Evaluates <i>operator</i> on every occurrence of <i>field</i> ; useful for passage retrieval
#filreq( c s )	Filter reject	Evaluate the expression <i>s</i> only if <i>c</i> is not satisfied
#filreq( c s )	Filter require	Evaluate the expression <i>s</i> only if <i>c</i> is satisfied

Table 1: Indri query language operators

Consider the following information need: “I want paragraphs from news feed articles published between 1991 and 2000 that mention a person, a monetary amount, and the company InfoCom.”

This need can be expressed in the following Indri query:

```
#filreq(
  #band( NewsFeeddoctype
    #date:between(1991 2000) )
  #combine[paragraph] (
    #any:person
    #any:money InfoCom ) )
```

Not bound by schema

e.g., don't need to know column names

use ML to select relevant documents

# An Introduction to Neural Information Retrieval

**Suggested Citation:** Bhaskar Mitra and Nick Craswell (2018), "An Introduction to Neural Information Retrieval", : Vol. xx, No. xx, pp 1–18. DOI: 10.1561/XXXXXXX.

**Bhaskar Mitra**  
Microsoft, University College London  
Montreal, Canada  
[bmitra@microsoft.com](mailto:bmitra@microsoft.com)

**Nick Craswell**  
Microsoft  
Bellevue, USA  
[nickcr@microsoft.com](mailto:nickcr@microsoft.com)

This article may be used only for the purpose of research, teaching, and/or private study. Commercial use or systematic downloading (by robots or other automatic processes) is prohibited without explicit Publisher approval.

**now**  
the essence of knowledge  
Boston — Delft

# What kind of ML do we use?

Today: whatever is hot in ML

- Neural nets
  - Vanilla deep networks
  - GANs
  - Attention networks
- Word embeddings

# An Introduction to Neural Information Retrieval

**Suggested Citation:** Bhaskar Mitra and Nick Craswell (2018), "An Introduction to Neural Information Retrieval", : Vol. xx, No. xx, pp 1–18. DOI: 10.1561/XXXXXXXXXX.

Bhaskar Mitra

Microsoft, University College London  
Montreal, Canada  
bmitra@microsoft.com

Nick Craswell

Microsoft  
Bellevue, USA  
nickcr@microsoft.com

## Improvements That Don't Add Up: Ad-Hoc Retrieval Results Since 1998

Timothy G. Armstrong, Alastair Moffat, William Webber, Justin Zobel  
Computer Science and Software Engineering  
The University of Melbourne  
Victoria 3010, Australia  
tgarm,alastair,webb,jz@csse.unimelb.edu.au

**ABSTRACT**  
The existence and use of standard test collections in information retrieval experimentation allows much to be compared between research groups and over time. Such comparisons, however, are rarely made. Most researchers only report results from their own experiments, a practice that allows a lack of overall improvement to go unnoticed. In this paper, we analyse results achieved at the TREC Ad-Hoc, Web, Text, and Robust collections as reported in SIGIR (1998–2008) and CIDEr (2004–2008). Dozens of individual and other claim statistically significant improvements; however, there is little evidence of improvement in ad-hoc retrieval technology over the past decade. Baseline performance is generally weak, often being at the mercy of the best TREC system. And while a handful of experiments find significant results, the vast majority succeed. Given this finding, we question the value of achieving ever statistically significant progress, or at least prevent the lack of it from going unnoticed. We describe an online database of retrieval runs that facilitate a practical and regular longitudinal comparison to ensure meaningful progress.

## Categories and Subject Descriptors

H.3.3 [Information Storage and Retrieval]: Systems and software—performance evaluation.  
General Terms

Experimentation, Measurement, Standardization.

Keywords

Retrieval experiment, evaluation, system measurement, survey.

1. INTRODUCTION

Information retrieval (IR) research has a strong tradition of empirical evaluation, stretching back to the Cranfield experiments of

1900 to make digital or hard copies of all or part of this work for personal use and internal distribution only. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission or a license from the copyright holder. © 2009 ACM 0-9824241-2-6, Hong Kong, China  
10.1145/1539356.1539371



# Bayesian Statistics Without Tears: A Sampling–Resampling Perspective

A. F. M. SMITH and A. E. GELFAND\*

Even to the initiated, statistical calculations based on Bayes's Theorem can be daunting because of the numerical integrations required in all but the simplest applications. Moreover, from a teaching perspective, introductions to Bayesian statistics—if they are given at all—are circumscribed by these apparent calculational difficulties. Here we offer a straightforward sampling–resampling perspective on Bayesian inference, which has both pedagogic appeal and suggests easily implemented calculation strategies.

**KEY WORDS:** Bayesian inference; Exploratory data analysis; Graphical methods; Influence; Posterior distribution; Prediction; Prior distribution; Random variate generation; Sampling–resampling techniques; Sensitivity analysis; Weighted bootstrap.

## 1. INTRODUCTION

Given data  $x$  obtained under a parametric model indexed by finite-dimensional  $\theta$ , the Bayesian learning process is based on

$$p(\theta|x) = \frac{l(\theta; x)p(\theta)}{\int l(\theta; x)p(\theta) d\theta}, \quad (1.1)$$

the familiar form of Bayes's Theorem, relating the posterior distribution  $p(\theta|x)$  to the likelihood  $l(\theta; x)$ , and the prior distribution  $p(\theta)$ . If  $\theta = (\phi, \psi)$ , with interest centering on  $\phi$ , the joint posterior distribution is marginalized to give the posterior distribution for  $\phi$ ,

$$p(\phi|x) = \int p(\phi, \psi|x) d\psi. \quad (1.2)$$

If summary inferences in the form of posterior expectations are required (e.g., posterior means and variances), these are based on

$$E[m(\theta)|x] = \int m(\theta)p(\theta|x) d\theta, \quad (1.3)$$

for suitable choices of  $m(\cdot)$ .

Thus, in the continuous case, the integration operation plays a fundamental role in Bayesian statistics, whether it is for calculating the normalizing constant in

(1.1), the marginal distribution in (1.2), or the expectation in (1.3). However, except in simple cases, explicit evaluation of such integrals will rarely be possible, and realistic choices of likelihood and prior will necessitate the use of sophisticated numerical integration or analytic approximation techniques (see, for example, Smith et al. 1985, 1987; Tierney and Kadane, 1986). This can pose problems for the applied practitioner seeking routine, easily implemented procedures. For the student, who may already be puzzled and discomfited by the intrusion of too much calculus into what ought surely to be a simple, intuitive, statistical learning process, this can be totally off-putting.

In the following sections, we address this problem by taking a new look at Bayes's Theorem from a sampling–resampling perspective. This will open the way to both easily implemented calculations and essentially calculus-free insight into the mechanics and uses of Bayes's Theorem.

## 2. FROM DENSITIES TO SAMPLES

As a first step, we note the essential duality between a sample and the density (distribution) from which it is generated. Clearly, the density generates the sample; conversely, given a sample we can approximately recreate the density (as a histogram, a kernel density estimate, an empirical cdf, or whatever).

Suppose we now shift the focus in (1.1) from densities to samples. In terms of densities, the inference process is encapsulated in the updating of the prior density  $p(\theta)$  to the posterior density  $p(\theta|x)$  through the medium of the likelihood function  $l(\theta; x)$ . Shifting to samples, this corresponds to the updating of a sample from  $p(\theta)$  to a sample from  $p(\theta|x)$  through the likelihood function  $l(\theta; x)$ .

In Section 3, we examine two resampling ideas that provide techniques whereby samples from one distribution may be modified to form samples from another distribution. In Section 4, we illustrate how these ideas may be utilized to modify prior samples to posterior samples, as well as to modify posterior samples arising under one model specification to posterior samples arising under another. An illustrative example is provided in Section 5.

## 3. TWO RESAMPLING METHODS

Suppose that a sample of random variates is easily generated, or has already been generated, from a continuous density  $g(\theta)$ , but that what is really required is a sample from a density  $h(\theta)$  absolutely continuous with

# What kind of ML do we use?

Then: probabilistic language models

- Famous, highly influential paper
- Bayesian approaches are quite old
- Needed hardware to catch up to actually be useful
- Peak Bayesian language modeling
  - 1962 – first year over 100 papers
  - Steady increase until 2008 ( $\sim 12k \rightarrow 24k$ )
  - Overall trend: still increasing, more fluctuation

(has "cooled off")

[https://app.dimensions.ai/discover/publication?search\\_mode=content&search\\_text=probabilistic%20language%20models%20&search\\_type=kws&search\\_field=full\\_search](https://app.dimensions.ai/discover/publication?search_mode=content&search_text=probabilistic%20language%20models%20&search_type=kws&search_field=full_search)

# Bayesian Statistics Without Tears: A Sampling–Resampling Perspective

A. F. M. SMITH and A. E. GELFAND\*

Even to the initiated, statistical calculations based on Bayes's Theorem can be daunting because of the numerical integrations required in all but the simplest applications. Moreover, from a teaching perspective, introductions to Bayesian statistics—if they are given at all—are circumscribed by these apparent calculational difficulties. Here we offer a straightforward sampling–resampling perspective on Bayesian inference, which has both pedagogic appeal and suggests easily implemented calculation strategies.

**KEY WORDS:** Bayesian inference; Exploratory data analysis; Graphical methods; Influence; Posterior distribution; Prediction; Prior distribution; Random variate generation; Sampling–resampling techniques; Sensitivity analysis; Weighted bootstrap.

## 1. INTRODUCTION

Given data  $x$  obtained under a parametric model indexed by finite-dimensional  $\theta$ , the Bayesian learning process is based on

$$p(\theta|x) = \frac{l(\theta; x)p(\theta)}{\int l(\theta; x)p(\theta) d\theta}, \quad (1.1)$$

the familiar form of Bayes's Theorem, relating the posterior distribution  $p(\theta|x)$  to the likelihood  $l(\theta; x)$ , and the prior distribution is  $p(\theta)$ . If  $\theta = (\phi, \psi)$ , with interest centering on  $\phi$ , the joint posterior distribution is marginalized to give the posterior distribution for  $\phi$ ,

$$p(\phi|x) = \int p(\phi, \psi|x) d\psi. \quad (1.2)$$

If summary inferences in the form of posterior expectations are required (e.g., posterior means and variances), these are based on

$$E[m(\theta)|x] = \int m(\theta)p(\theta|x) d\theta, \quad (1.3)$$

for suitable choices of  $m(\cdot)$ .

Thus, in the continuous case, the integration operation plays a fundamental role in Bayesian statistics, whether it is for calculating the normalizing constant in

(1.1), the marginal distribution in (1.2), or the expectation in (1.3). However, except in simple cases, explicit evaluation of such integrals will rarely be possible, and realistic choices of likelihood and prior will necessitate the use of sophisticated numerical integration or analytic approximation techniques (see, for example, Smith et al. 1985, 1987; Tierney and Kadane, 1986). This can pose problems for the applied practitioner seeking routine, easily implemented procedures. For the student, who may already be puzzled and discomfited by the intrusion of too much calculus into what ought surely to be a simple, intuitive, statistical learning process, this can be totally off-putting.

In the following sections, we address this problem by taking a new look at Bayes's Theorem from a sampling–resampling perspective. This will open the way to both easily implemented calculations and essentially calculus-free insight into the mechanics and uses of Bayes's Theorem.

## 2. FROM DENSITIES TO SAMPLES

As a first step, we note the essential duality between a sample and the density (distribution) from which it is generated. Clearly, the density generates the sample; conversely, given a sample we can approximately recreate the density (as a histogram, a kernel density estimate, an empirical cdf, or whatever).

Suppose we now shift the focus in (1.1) from densities to samples. In terms of densities, the inference process is encapsulated in the updating of the prior density  $p(\theta)$  to the posterior density  $p(\theta|x)$  through the medium of the likelihood function  $l(\theta; x)$ . Shifting to samples, this corresponds to the updating of a sample from  $p(\theta)$  to a sample from  $p(\theta|x)$  through the likelihood function  $l(\theta; x)$ .

In Section 3, we examine two resampling ideas that provide techniques whereby samples from one distribution may be modified to form samples from another distribution. In Section 4, we illustrate how these ideas may be utilized to modify prior samples to posterior samples, as well as to modify posterior samples arising under one model specification to posterior samples arising under another. An illustrative example is provided in Section 5.

## 3. TWO RESAMPLING METHODS

Suppose that a sample of random variates is easily generated, or has already been generated, from a continuous density  $g(\theta)$ , but that what is really required is a sample from a density  $h(\theta)$  absolutely continuous with

# What is a language model?

Probabilistic model that takes text input:

$P(\text{query} | \text{document})$

$P(\text{query} | \text{passage})$

$P(\text{tag} | \text{document})$

We use the LM for prediction, classification, etc.

\*A. F. M. Smith is Professor, Department of Mathematics, Imperial College of Science Technology and Medicine, London SW7 2BZ, England. A. E. Gelfand is Professor, Department of Statistics, University of Connecticut, Storrs, CT 06269. The authors are grateful to David Stephens for assistance with computer experiments. His work and a visit to the United Kingdom by the second author were supported by the U.K. Science and Engineering Council Complex Stochastic Systems Initiative.

# Bayesian Statistics Without Tears: A Sampling–Resampling Perspective

A. F. M. SMITH and A. E. GELFAND\*

Even to the initiated, statistical calculations based on Bayes's Theorem can be daunting because of the numerical integrations required in all but the simplest applications. Moreover, from a teaching perspective, introductions to Bayesian statistics—if they are given at all—are circumscribed by these apparent calculational difficulties. Here we offer a straightforward sampling–resampling perspective on Bayesian inference, which has both pedagogic appeal and suggests easily implemented calculation strategies.

**KEY WORDS:** Bayesian inference; Exploratory data analysis; Graphical methods; Influence; Posterior distribution; Prediction; Prior distribution; Random variate generation; Sampling–resampling techniques; Sensitivity analysis; Weighted bootstrap.

## 1. INTRODUCTION

Given data  $x$  obtained under a parametric model indexed by finite-dimensional  $\theta$ , the Bayesian learning process is based on

$$p(\theta|x) = \frac{l(\theta; x)p(\theta)}{\int l(\theta; x)p(\theta) d\theta}, \quad (1.1)$$

the familiar form of Bayes's Theorem, relating the posterior distribution  $p(\theta|x)$  to the likelihood  $l(\theta; x)$ , and the prior distribution is  $p(\theta)$ . If  $\theta = (\phi, \psi)$ , with interest centering on  $\phi$ , the joint posterior distribution is marginalized to give the posterior distribution for  $\phi$ ,

$$p(\phi|x) = \int p(\phi, \psi|x) d\psi. \quad (1.2)$$

If summary inferences in the form of posterior expectations are required (e.g., posterior means and variances), these are based on

$$E[m(\theta)|x] = \int m(\theta)p(\theta|x) d\theta, \quad (1.3)$$

for suitable choices of  $m(\cdot)$ .

Thus, in the continuous case, the integration operation plays a fundamental role in Bayesian statistics, whether it is for calculating the normalizing constant in

(1.1), the marginal distribution in (1.2), or the expectation in (1.3). However, except in simple cases, explicit evaluation of such integrals will rarely be possible, and realistic choices of likelihood and prior will necessitate the use of sophisticated numerical integration or analytic approximation techniques (see, for example, Smith et al. 1985, 1987; Tierney and Kadane, 1986). This can pose problems for the applied practitioner seeking routine, easily implemented procedures. For the student, who may already be puzzled and discomfited by the intrusion of too much calculus into what ought surely to be a simple, intuitive, statistical learning process, this can be totally off-putting.

In the following sections, we address this problem by taking a new look at Bayes's Theorem from a sampling–resampling perspective. This will open the way to both easily implemented calculations and essentially calculus-free insight into the mechanics and uses of Bayes's Theorem.

## 2. FROM DENSITIES TO SAMPLES

As a first step, we note the essential duality between a sample and the density (distribution) from which it is generated. Clearly, the density generates the sample; conversely, given a sample we can approximately recreate the density (as a histogram, a kernel density estimate, an empirical cdf, or whatever).

Suppose we now shift the focus in (1.1) from densities to samples. In terms of densities, the inference process is encapsulated in the updating of the prior density  $p(\theta)$  to the posterior density  $p(\theta|x)$  through the medium of the likelihood function  $l(\theta; x)$ . Shifting to samples, this corresponds to the updating of a sample from  $p(\theta)$  to a sample from  $p(\theta|x)$  through the likelihood function  $l(\theta; x)$ .

In Section 3, we examine two resampling ideas that provide techniques whereby samples from one distribution may be modified to form samples from another distribution. In Section 4, we illustrate how these ideas may be utilized to modify prior samples to posterior samples, as well as to modify posterior samples arising under one model specification to posterior samples arising under another. An illustrative example is provided in Section 5.

## 3. TWO RESAMPLING METHODS

Suppose that a sample of random variates is easily generated, or has already been generated, from a continuous density  $g(\theta)$ , but that what is really required is a sample from a density  $h(\theta)$  absolutely continuous with

# What is tf-idf?

tf-idf: “term frequency – inverse document frequency”

- statistical, not probabilistic
- old and still shockingly good

\*A. F. M. Smith is Professor, Department of Mathematics, Imperial College of Science Technology and Medicine, London SW7 2BZ, England. A. E. Gelfand is Professor, Department of Statistics, University of Connecticut, Storrs, CT 06269. The authors are grateful to David Stephens for assistance with computer experiments. His work and a visit to the United Kingdom by the second author were supported by the U.K. Science and Engineering Council Complex Stochastic Systems Initiative.

# Bayesian Statistics Without Tears: A Sampling–Resampling Perspective

A. F. M. SMITH and A. E. GELFAND\*

Even to the initiated, statistical calculations based on Bayes's Theorem can be daunting because of the numerical integrations required in all but the simplest applications. Moreover, from a teaching perspective, introductions to Bayesian statistics—if they are given at all—are circumscribed by these apparent calculational difficulties. Here we offer a straightforward sampling–resampling perspective on Bayesian inference, which has both pedagogic appeal and suggests easily implemented calculation strategies.

**KEY WORDS:** Bayesian inference; Exploratory data analysis; Graphical methods; Influence; Posterior distribution; Prediction; Prior distribution; Random variate generation; Sampling–resampling techniques; Sensitivity analysis; Weighted bootstrap.

## 1. INTRODUCTION

Given data  $x$  obtained under a parametric model indexed by finite-dimensional  $\theta$ , the Bayesian learning process is based on

$$p(\theta|x) = \frac{l(\theta; x)p(\theta)}{\int l(\theta; x)p(\theta) d\theta}, \quad (1.1)$$

the familiar form of Bayes's Theorem, relating the posterior distribution  $p(\theta|x)$  to the likelihood  $l(\theta; x)$ , and the prior distribution is  $p(\theta)$ . If  $\theta = (\phi, \psi)$ , with interest centering on  $\phi$ , the joint posterior distribution is marginalized to give the posterior distribution for  $\phi$ ,

$$p(\phi|x) = \int p(\phi, \psi|x) d\psi. \quad (1.2)$$

If summary inferences in the form of posterior expectations are required (e.g., posterior means and variances), these are based on

$$E[m(\theta)|x] = \int m(\theta)p(\theta|x) d\theta, \quad (1.3)$$

for suitable choices of  $m(\cdot)$ .

Thus, in the continuous case, the integration operation plays a fundamental role in Bayesian statistics, whether it is for calculating the normalizing constant in

(1.1), the marginal distribution in (1.2), or the expectation in (1.3). However, except in simple cases, explicit evaluation of such integrals will rarely be possible, and realistic choices of likelihood and prior will necessitate the use of sophisticated numerical integration or analytic approximation techniques (see, for example, Smith et al. 1985, 1987; Tierney and Kadane, 1986). This can pose problems for the applied practitioner seeking routine, easily implemented procedures. For the student, who may already be puzzled and discomfited by the intrusion of too much calculus into what ought surely to be a simple, intuitive, statistical learning process, this can be totally off-putting.

In the following sections, we address this problem by taking a new look at Bayes's Theorem from a sampling–resampling perspective. This will open the way to both easily implemented calculations and essentially calculus-free insight into the mechanics and uses of Bayes's Theorem.

## 2. FROM DENSITIES TO SAMPLES

As a first step, we note the essential duality between a sample and the density (distribution) from which it is generated. Clearly, the density generates the sample; conversely, given a sample we can approximately recreate the density (as a histogram, a kernel density estimate, an empirical cdf, or whatever).

Suppose we now shift the focus in (1.1) from densities to samples. In terms of densities, the inference process is encapsulated in the updating of the prior density  $p(\theta)$  to the posterior density  $p(\theta|x)$  through the medium of the likelihood function  $l(\theta; x)$ . Shifting to samples, this corresponds to the updating of a sample from  $p(\theta)$  to a sample from  $p(\theta|x)$  through the likelihood function  $l(\theta; x)$ .

In Section 3, we examine two resampling ideas that provide techniques whereby samples from one distribution may be modified to form samples from another distribution. In Section 4, we illustrate how these ideas may be utilized to modify prior samples to posterior samples, as well as to modify posterior samples arising under one model specification to posterior samples arising under another. An illustrative example is provided in Section 5.

## 3. TWO RESAMPLING METHODS

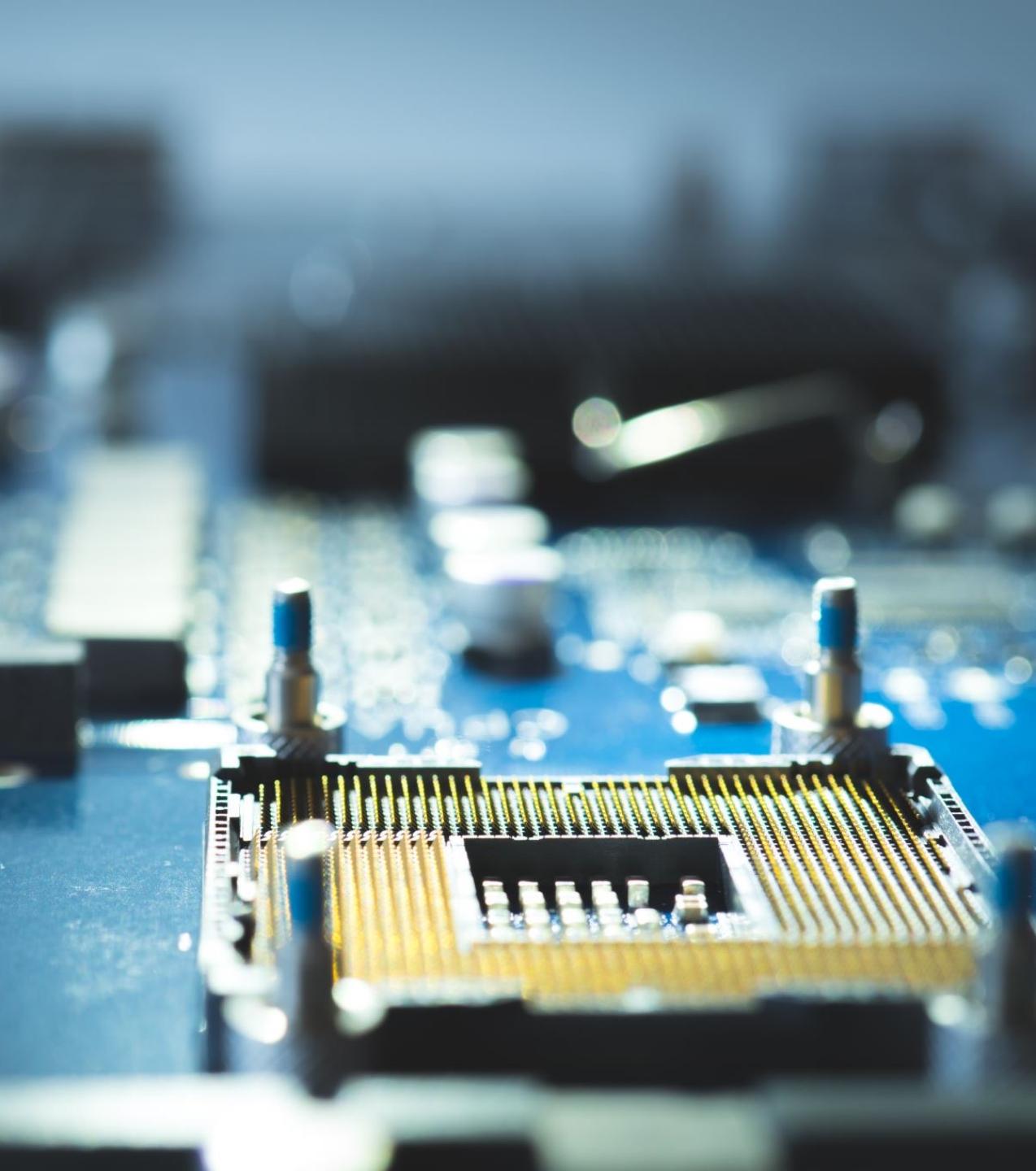
Suppose that a sample of random variates is easily generated, or has already been generated, from a continuous density  $g(\theta)$ , but that what is really required is a sample from a density  $h(\theta)$  absolutely continuous with

# Bayesian modeling

## Important concepts:

- “Bayesian” is more than Bayes Rule
- Belief vs. probability
- Joint vs. conditional probabilities

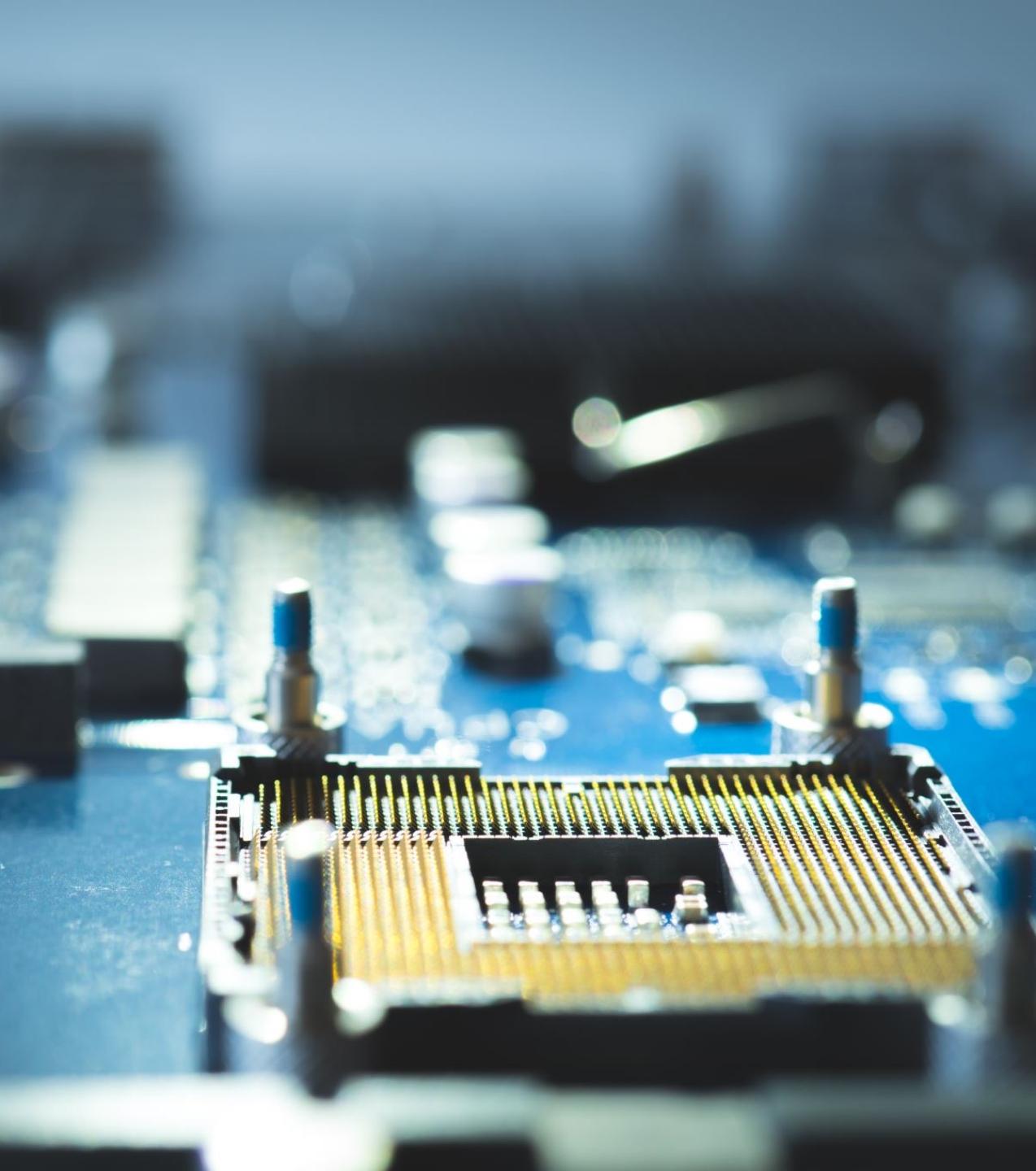
\*A. F. M. Smith is Professor, Department of Mathematics, Imperial College of Science Technology and Medicine, London SW7 2BZ, England. A. E. Gelfand is Professor, Department of Statistics, University of Connecticut, Storrs, CT 06269. The authors are grateful to David Stephens for assistance with computer experiments. His work and a visit to the United Kingdom by the second author were supported by the U.K. Science and Engineering Council Complex Stochastic Systems Initiative.



# Bayesian modeling

Important concepts:

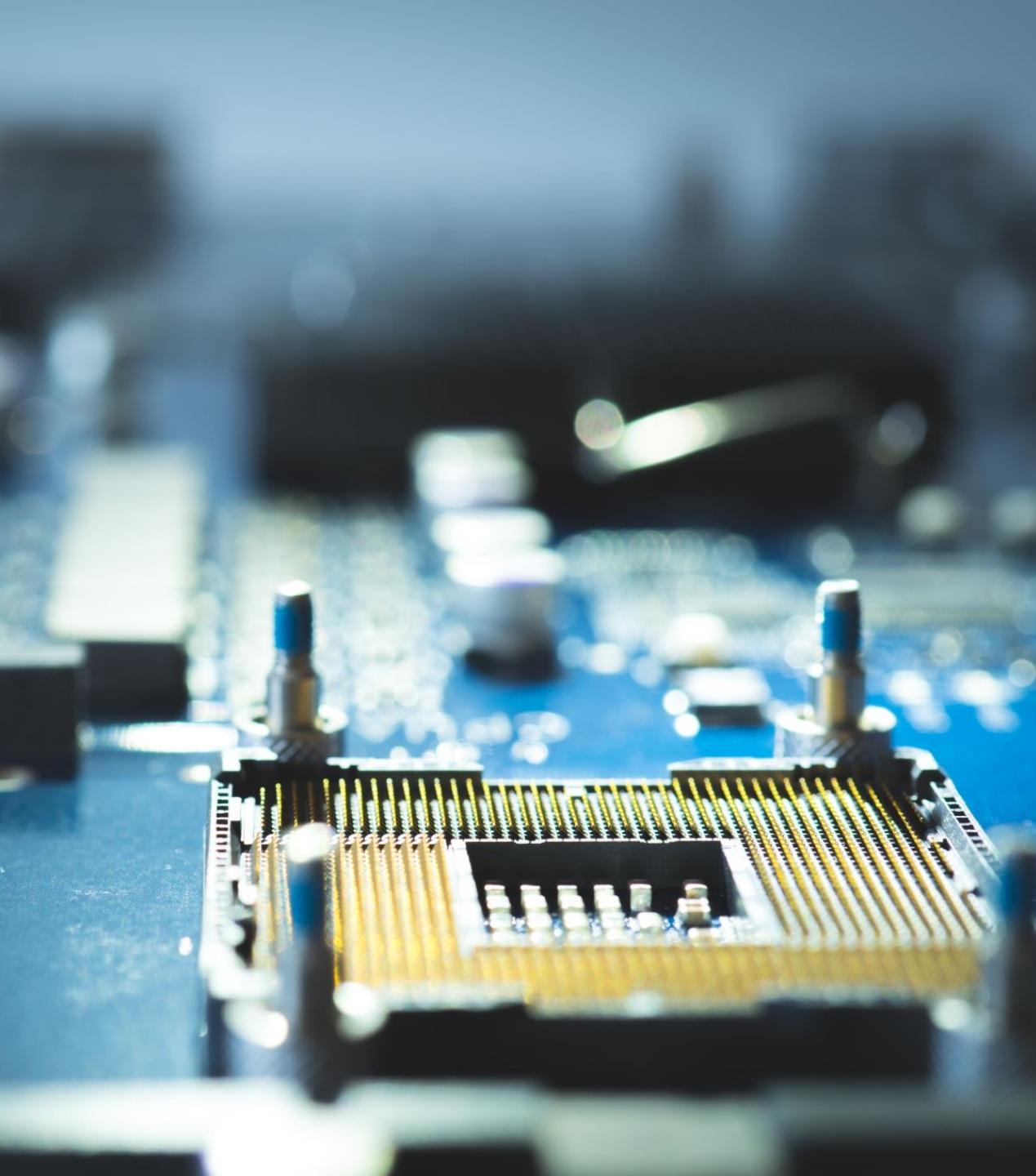
- “**Bayesian**” is more than **Bayes Rule**
  - Frequentist – counts
    - Philosophical – some underlying true phenomenon
    - Assumptions – constant or equally likely
  - Bayesian -- belief
    - Easier to encode domain-specific knowledge



# Bayesian modeling

Important concepts:

- “Bayesian” is more than Bayes Rule
- **Belief vs. Probability**
  - Belief encodes possible worlds



# Bayesian modeling

Important concepts:

- “Bayesian” is more than Bayes Rule
- Belief vs. Probability
- **Joint vs. conditional probabilities**
  - Posterior is may not have closed form
  - May still need to marginalize



## VERMONT ADVANCED COMPUTING CORE

Who here has had a job time out, cancelled, or fail on them?

# Why this all might matter to you

- Using slurm directly = pre-MapReduce
  - Benefits: 100% freedom
  - Not tied to tech. or data format

The screenshot shows the homepage of the Vermont Advanced Computing Core. It features a large blue starburst graphic containing the question "Who here has had a job time out, cancelled, or fail on them?". To the left of the starburst are several service icons and descriptions:

- HPCC**: Three high-performance clusters (BlackDiamond, Bluemoon, Borealis) for computation, low-latency throughput.
- VACC**: Various accounts.
  - VACC account help
  - UVM faculty accounts
- Services**: Onboarding for those new to performance computing.
- SchedMD**: A link to the SchedMD platform.

Below these are sections for "Dashboard" (for account owners), "Documentation", and "Enhanced by Google". A tweet from the official account (@uvmvacc) is also visible on the right side of the page.



## Documentation

**NOTE:** This documentation is for Slurm version 21.08.

Documentation for older versions of Slurm are distributed with the source, or may be found in the



MENU

## VERMONT ADVANCED COMPUTING CORE



## AMD GIVES UVM \$1 MILLION TO BOOST COMPUTING POWER FOR COVID-19 RESEARCH

UVM is among the first 21 schools the company has supported through its AMD HPC Fund for COVID-19 Research program.

[Read more about this gift.](#)



HPC

Three high performance computing (HPC) clusters – BlackDiamond, Bluemoon, DeepGreen – which support large-scale computation, low-latency networking for MPI workloads, and high-throughput AI and machine learning workflows.



## Data Management

Various data storage plans to meet the needs of our:

- VACC account holders
- UVM faculty



## Services

Onboarding for those new to high performance computing

## JOIN THE VACC

## Cost / Payment

VACC account holders join one of three tiers on a yearly basis.

## Request Account

Principal Investigators (PIs) or IT support working with PIs may request an account.

## USE THE VACC

## Knowledge Base

Help topics include connecting to the cluster, moving files, running a job, and more.

## Dashboard

For account owners (PIs), [not sponsored users](#)

## Tweets by @uvmvacc

Vermont Advanced Computi  
@uvmvacc

Bluemoon cluster about to bring online  
5000 new compute cores!!!



Vermon Advanced Computi  
@uvmvacc

We're thrilled to receive funding from  
NSF for our cluster!

SchedMD

ENHANCED BY Google



## Documentation

NOTE: This documentation is for Slurm version  
21.08.

Documentation for older versions of Slurm are  
distributed with the source, or may be found in the

## Why this all might matter to you

- Using slurm directly = pre-MapReduce
  - Benefits: 100% freedom
  - Not tied to tech. or data format
- Costs: Must roll your own
  - Composing tasks? Probably manual
  - Need to serialize? Probably manual