Introduction to Spark 2.0 for Database Researchers

Michael Armbrust[1], Doug Bateman[1], Reynold Xin[1], Matei Zaharia[1,2]
[1]Databricks, [2] MIT CSAIL
{michael, doug.bateman, rxin, matei}@databricks.com

1. INTRODUCTION

Originally started as an academic research project at UC Berkeley, Apache Spark is one of the most popular open source projects for big data analytics. Over 1000 volunteers have contributed code to the project; it is supported by virtually every commercial vendor; many universities are now offering courses on Spark.

Spark has evolved significantly since the 2010 research paper: its foundational APIs are becoming more relational and structural with the introduction of the Catalyst relational optimizer, and its execution engine is developing quickly to adopt the latest research advances in database systems such as whole-stage code generation.

This tutorial is designed for database researchers (graduate students, faculty members, and industrial researchers) interested in a brief hands-on overview of Spark. This tutorial covers the core APIs for using Spark 2.0, including DataFrames, Datasets, SQL, streaming and machine learning pipelines. Each topic includes slide and lecture content along with hands-on use of a Spark cluster through a web-based notebook environment.

In addition, we will dive into the engine internals to discuss architectural design choices and their implications in practice. We will guide the audience to "hack" Spark by extending its query optimizer to speed up distributed join execution.

2. TARGET AUDIENCE

The target audience for this tutorial is anyone with an interest in Spark and more generally big data processing. The audience is expected to have a basic understanding of database systems, and bring a laptop with Chrome/Safari browser in order to complete the hands-on exercise.

Upon completion of the tutorial, the audience should walk away with a good understanding of the state-of-the-art in Spark in order to build research projects or create educational materials.

3. TUTORIAL OUTLINE

This tutorial is divided into four parts. We first introduce Spark 2.0 and the main programming abstraction it offers, followed by discussions on engine internals. A large fraction of the tutorial features hands-on coding exercises.

SIGMOD '16, June 26–July 1, 2016, San Francisco, CA, USA.

ACM 978-1-4503-3531-7/16/06…$15.00
DOI: http://dx.doi.org/10.1145/2882903.2912565

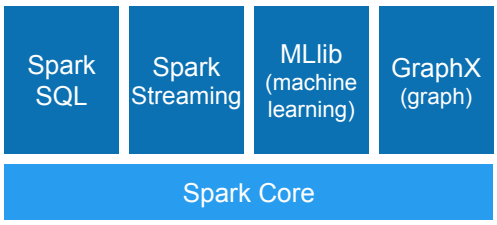

Figure 1: Spark software stack, showing the various libraries implemented over the core engine.

3.1 Spark from an end-user's perspective

We will first introduce Spark, its history, and many of the projects in the Spark ecosystem [6, 3, 5, 1, 4, 2].

We will then focus on the main APIs in Spark 2.0, including the typed Dataset API and the untyped DataFrame API for both interactive data analysis and production data engineering. This module comes with hands-on exercise in the Databricks cloud Spark platform using notebooks.

3.2 Catalyst query optimizer

Catalyst is an extensible optimizer in Spark that builds on the functional programming constructs in Scala. CatalystâĂŹs extensible design had two purposes. First, we wanted to make it easy to add new optimization techniques and features to Spark SQL, especially for the purpose of tackling various problems we were seeing with big data (e.g., semistructured data and advanced analytics). Second, we wanted to enable external developers to extend the optimizer. For example, by adding data source specific rules that can push filtering or aggregation into external storage systems, or support for new data types. Catalyst supports both rule-based and cost-based optimization.

In this module, we will explain the internals of the Catalyst optimizer, and conduct a hands-on exercise to enhance the optimizer to conduct more efficient range joins.

3.3 Streaming and machine learning

This module covers two additional components in Spark: streaming and machine learning.

Structured Streaming is a new stream processing framework in Spark 2.0. It allows users to specify stream processing logic in the same way as batch analytics, and the engine leverages the Catalyst query optimizer to automatically incrementalize the execution on streams.

MLlib is an open-source distributed machine learning library built on Spark. MLlib provides efficient functionality for a wide range of learning settings and includes several underlying statis-

tical, optimization, and linear algebra primitives. Shipped with Spark, MLlib supports several languages and provides a high-level API that leverages SparkâĂŹs rich ecosystem to simplify the development of end-to-end machine learning pipelines.

3.4 Engine internals

This module dives into the execution engine internals, covering some of the recent development in Spark, including runtime code generation, memory management, and scheduling.

4. PRESENTERS - BIOS

Michael Armbrust leads the development of Spark SQL, a project he created in 2014 that became the most popular component of Spark. He received his PhD from UC Berkeley in 2013, and was advised by Michael Franklin, David Patterson, and Armando Fox. His thesis focused on building systems that allow developers to rapidly build scalable interactive applications, and specifically defined the notion of scale independence. His interests broadly include distributed systems, large-scale structured storage and query optimization.

Doug Bateman is the Director of Training at Databricks and teaches advanced Spark classes for Databricks' customers. He has taught 800+ classes on Java, Spring, Hibernate, Python, and Android and 15+ years of software engineering experience.

Reynold Xin is a cofounder and Chief Architect for Spark at Databricks. Prior to Databricks, he was pursuing a PhD at the UC Berkeley AMPLab, advised by Michael Franklin. While at Berkeley, he worked on CrowdDB, Shark and GraphX. He also set the 2014 world record in 100 TB sorting (Daytona Gray), beating the previous 2013 Hadoop record by 30X on per-node efficiency.

Matei Zaharia is an assistant professor of computer science at MIT and CTO of Databricks, the company commercializing Apache Spark. He started the Spark project during his PhD at UC Berkeley. He is broadly interested in large-scale computer systems and networks, and has also contributed to projects including Mesos, Hadoop, Tachyon and Shark. Matei received the 2014 ACM Doctoral Dissertation award for his research on Spark, as well as best paper awards at NSDI and SIGCOMM.

Acknowledgments: We would like to thank Alexandros Labrinidis and Sam Madden for their feedback and help with this tutorial.

5. REFERENCES

[1] M. Armbrust, T. Das, A. Davidson, A. Ghodsi, A. Or, J. Rosen, I. Stoica, P. Wendell, R. Xin, and M. Zaharia. Scaling spark in the real world: performance and usability. *Proceedings of the VLDB Endowment*, 8(12):1840–1843, 2015.

[2] M. Armbrust, R. S. Xin, C. Lian, Y. Huai, et al. Spark SQL: Relational data processing in Spark. In *SIGMOD*, pages 1383–1394, 2015.

[3] J. E. Gonzalez, R. S. Xin, A. Dave, D. Crankshaw, M. J. Franklin, and I. Stoica. Graphx: Graph processing in a distributed dataflow framework. In *OSDI 2014*, pages 599–613.

[4] X. Meng, J. Bradley, B. Yavuz, E. Sparks, S. Venkataraman, D. Liu, J. Freeman, D. Tsai, M. Amde, S. Owen, et al. Mllib: Machine learning in apache spark. *arXiv preprint arXiv:1505.06807*, 2015.

[5] S. Venkataraman, Z. Yang, E. L. Davies Liu, H. Falaki, X. Meng, R. Xin, A. Ghodsi, M. Franklin, I. Stoica, and M. Zaharia. Sparkr: Scaling r programs with spark. In *SIGMOD*, 2016.

[6] R. S. Xin, J. Rosen, M. Zaharia, M. J. Franklin, S. Shenker, and I. Stoica. Shark: SQL and rich analytics at scale. In *SIGMOD 2013*.