Adaptive Consistency Management for Distributed Machine Learning

by

Christoph Alt

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Computer Science and Engineering

at the

TECHNISCHE UNIVERSITÄT BERLIN

March 2017

Author . . . .
Department of Electrical Engineering and Computer Science
March 15, 2017

Certified by . . . .
Prof. Dr. Odej Kao
Associate Professor
Thesis Supervisor

Accepted by . . . .
Prof. Dr. Voker Markl

Adaptive Consistency Management for Distributed Machine Learning

by

Christoph Alt

Submitted to the Department of Electrical Engineering and Computer Science
on March 15, 2017, in partial fulfillment of the
requirements for the degree of
Master of Science in Computer Science and Engineering

Abstract

In recent years, machine learning emerged as an essential part of many successful applications and businesses. The ever growing amount of data requires these algorithms, many of them sequential in nature, to be executed in a distributed manner on a large scale. This is commonly achieved by exploiting the algorithms stochastic nature to allow for parallel execution at the expense of lowered consistency among the distributed data structures. Even though the quality of the result is not affected, an improper level of consistency can severely affect algorithm performance, resulting in a non optimal convergence rate. Furthermore the level of consistency required to achieve the best performance can change during different periods of algorithm execution. System architecture, topology, algorithm, hardware and data properties influence performance as well. Despite its widespread use, the implications of these aspects of distributed systems on algorithm performance are not yet well understood. This thesis aims to answer the question of how the level of consistency affects the overall performance of distributed machine learning algorithms and also aims to identify strategies for consistency management that can help to mitigate the negative effect on performance. Based on the identified strategies, a protocol will be developed that is able to manage the consistency level of distributed state. The protocol will be designed in a way that allows for dynamic changes in consistency. This enables the possibility of adaptive consistency management in distributed machine learning environments based on heuristics and algorithm progress.

Thesis Supervisor: Prof. Dr. Odej Kao
Title: Associate Professor

Acknowledgments

This is the acknowledgements section. You should replace this with your own acknowledgements.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

One of the most challenging tasks in computer science and engineering resolves around improving performance of a computational system. Whether this has been done by making the hardware faster or inventing new systems, strategies and algorithms to parallelize work more efficiently. Machine learning is no exception and efficient parallelization is a key aspect towards more intelligent systems. Over the past decade the need for processing large amounts of data increased rapidly. Many companies and institutions discovered the value inherent to data and started collecting vast amounts, e.g. user activities or sensor data. The amount of data to be processed by a single algorithm is currently somewhere in the range between 100 gigabytes to tenth of petabytes. While in the past, processing of data that could not be handled by a single machine was often reserved to organizations owning a supercomputer and therefore of limited use, recent developments in frameworks for distributed computing and the increase in cost efficiency of hardware changed this. The success of what is now commonly refered to as "Big Data" started with the introduction of MapReduce [1] in 2004. The framework made it possible to process data in a distributed and fault tolerant way even though with the help of a compute cluster consisting of hundredth or thousandth of machines. Instead of using a single, expensive, special hardware supercomputer, the framework provides the ability to combine commodity hardware machines into a compute cluster. The framework then takes care of all necessary aspects to ensure a fault tolerant and parallel execution of the task submitted to the cluster. The advantage over previous approachs is that the framework could be run entirely on top of machines using commodity hardware. Which does not require special hardware and therefore equals low cost. Since then, many of today's successful companies rely heavily on the ability to process vast amounts of data to improve their services and customer experience and it is used throughout many diverse fields ranging from finance, commerce all the way to healthcare and many more. This alone is a challenge by itself and over the past years many new frameworks have been introduced to make development and execution of data mining and machine learning algorithms considerably easier.

1.1 MapReduce and Beyond

Google's MapReduce has esentially lead the path to a widespread adoption of big data processing and the increased awareness of the value of big data. The MapReduce has been adapted by the Apache Hadoop project [2], which quickly gained widespread adoption and by now provides a whole ecosystem around big data processing. Including a fault tolerant distributed filesystem (HDFS), MapReduce framework and a resource manager (YARN) [3]. On the other hand, MapReduce had some practical limitations that lead to the invention of more sophisticated and specialised big data frameworks, with the most widely used frameworks such as Apache Spark [4], Apache Flink [5] and GraphLab [6].

Google introduced its MapReduce system in 200x (c), with Apache Hadoop, Apache Spark (c), Apache Flink (c) and GraphLab (c) being the most widely used frameworks by the time. The first three frameworks are based upon the data-flow paradigm (c), whereas the latter uses a graph abstraction to model particular algorithms. While the primary focus of these systems lies on ETL and graph processing, they are often used to run machine learning algorithms as well. While this works well for algorithms that can be parallelized efficiently, performance on algorithms requiring many fast and probably asynchronous iterations to refine a model is not satisfying (c)(such as Logistic Regression or Latent Dirichlet Allocation). This fact led to the development of specialized frameworks, such as Petuum (c), ParameterServer (c) or Yahoo!LDA. These frameworks inhabit a completely different paradigm, not based on dataflow. While these frameworks are well suited for their particular purpose, namely ETL (c, fn) and execution of graph algorithms, when it comes to machine learning, these systems have some disadvantages, while these systems are very good at embarrassingly parallel algorithms and namely the which led to poor performance when executing algorithms that quickly execute iterative model refinements (such as Logistic Regression) (often referred to as iterative convergent algorithms) due to large iteration overhead and the lack of executing iterations asynchronously on different nodes, which led to the invention of new systems. While these systems often increase the performance on machine learning algorithms by an order of magnitude compared to dataflow systems, most systems come with either limited usability which makes it difficult to implement additional algorithms, are tied to a specific algorithm or are very low level frameworks. Most recognized frameworks are Petuum (c), ParameterServer (c) and Yahoo!LDA (c). During the past couple of years the amount of data to be processed by machine learning algorithms has increased steadily. Machine learning has become the core of many successful businesses and continues to play an important role in the artificial intelligence community. Making products smarter and tied to users needs. Since the introduction of Google MapReduce a lot of new frameworks for large scale data mining have been introduced to make the processing of vast amounts of data more easier and more efficient. Relying on the capability to process vast amount of data in a constrained setup (time, space). Data collected has immense value. Efficiently distributing machine learning algoritms remains and extremely challenging problem. This motivates our work in the area of distributed machine learning. Ease of implementing new algorithms,

1.2 Distributed Machine Learning

Machine learning has gained a lot of attention lately because it enables the possibility to learn from the collected data and works best with lot of data to learn from. Which is given by the vast amoutn of data collected. Improving speed, allowing more degrees of freedom (async program flow), ease the development of algorithms close to single machine implementations, compiler, letting the developer and researcher choose the strategy if necessary or implement heuristics <- challenge and motivation

1.3 Thesis Outline

introducing a novel system for distributed machine learning and researching mechanism to dynamically adapt and manage consistency among distributed state describe each chapter

In this thesis we introduce a novel system for large scale distributed machine learning, which

Chapter 2

Background

2.1 Iterative Convergent Algorithms

2.2 Dataflow Systems

2.3 Distributed Machine Learning

Chapter 3

State Centric Programming Model

Chapter 4

Consistency Management

Chapter 5

Experiments

Chapter 6

Conclusion

Bibliography

[1] J. Dean and S. Ghemawat, “MapReduce: Simplied Data Processing on Large Clusters,” *Proceedings of 6th Symposium on Operating Systems Design and Implementation*, pp. 137–149, 2004.

[2] A. Hadoop, “Hadoop,” 2009.

[3] V. Kumar Vavilapalli, A. C. Murthy, C. Douglas, S. Agarwal, M. Konar, R. Evans, T. Graves, J. Lowe, H. Shah, S. Seth, B. Saha, C. Curino, O. O ’malley, S. Radia, B. Reed, and E. Baldeschwieler, “Apache Hadoop YARN: Yet Another Resource Negotiator,” *SOCC ’13 Proceedings of the 4th annual Symposium on Cloud Computing*, vol. 13, pp. 1–3, 2013.

[4] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, “Spark : Cluster Computing with Working Sets,” *HotCloud’10 Proceedings of the 2nd USENIX conference on Hot topics in cloud computing*, p. 10, 2010.

[5] A. Alexandrov, R. Bergmann, S. Ewen, J. C. Freytag, F. Hueske, A. Heise, O. Kao, M. Leich, U. Leser, V. Markl, F. Naumann, M. Peters, A. Rheinländer, M. J. Sax, S. Schelter, M. Höger, K. Tzoumas, and D. Warneke, “The Stratosphere platform for big data analytics,” *VLDB Journal*, vol. 23, no. 6, pp. 939–964, 2014.

[6] Y. Low, J. Gonzalez, A. Kyrola, D. Bickson, and C. Guestrin, “GraphLab: A Distributed Framework for Machine Learning in the Cloud,” *The 38th International Conference on Very Large Data Bases*, vol. 5, no. 8, pp. 716–727, 2012.