Reconfigurable Machine Learning

Andrew Chung and Michael Kuchnik
Carnegie Mellon University

Abstract

Spot markets on cloud services provide a means of trading reliability for cheap computation. This paper addresses elasticity and scalability issues in a dynamic environment. A system architecture and implementation are described targeting rapid reconfiguration for optimizing the system based on its scale as machines are added and removed.

1 Introduction

Elastic machine learning has the flexibility to scale up and down as machines are made available. The spot market found on Google Computer Enginer (GCE) and Amazon Web Surface (AWS) allows users to purchase unreliable compute for a fraction of the cost. The discount comes at a price, however. Tasks can be evicted from spot instances as the demand for spot instances increase, drawing a parallel to machine failure. Previous work on TierML [8] demonstrated a system which can take advantage of the addition and eviction of transient resources. TierML leverages the LazyTable parameter server system [5]. In LazyTable, parameters are architected in a tabular manner with rows being user-defined data types for the machine learning job, and tables consisting of multiple rows.

The work in this paper attempts to improve on the work done with TierML. TierML presents three configurations (stages) of operation to deal with varying scale of the cluster, provided a set of reliable and transient nodes. Although the addition and removal of workers is supported, TierML is unable to reconfigure itself after launching. This means that changing the scale of the system through adding or removing large amounts of machines has the capability of placing the system in a less than optimal configuration. Even though TierML's main strength is in its elasticity, with a static configuration, TierML cannot truly demonstrate the benefits of it being able to scale up and down. In this paper, we attempt to extend TierML with capabilities to dynamically reconfigure itself in order to exhibit its full promise on elastic computing.

This paper presents the following contributions to the TierML system:

1. A design and implementation of protocols for reconfiguration of the TierML system (stage switching)
2. A flexible max-flow based model considering system bottleneck parameters that generate a recommended topology (stage) for the TierML system
3. Evaluation on the cost and benefits of stage switching and its protocols

2 Background

2.1 The Parameter Server Architecture and LazyTable

In the classic parameter server architecture for machine learning, there are two types of logical instances: the worker and the parameter server. Workers perform the actual machine learning workloads while parameter servers act as a specialized distributed key-value store for machine learning parameters [5].

2.2 TierML

An extension on LazyTable named TierML allows LazyTable to be run on a combination of on-demand (reliable) and spot instances in EC2.

TierML introduces the following four types of logical instances:

- **Backup Parameter Server (BackupPS)**: A root-level parameter server that resides on a reliable machine. Parameter servers run on on-demand instances that serve as a reliable backup to the ActivePS. The system assumes that the machine never experiences revocation and failures.
- **Active Parameter Server (ActivePS)**: Parameter servers that act as a “proxy" to BackupPS's. Serve clients and aggregate parameters to the BackupPS. Run on machines that can be revoked.
- **Worker**: Logical instances that perform the actual machine learning computation over sharded data fetched from a Distributed File System. Sends updates to its parent parameter server (ActivePS or BackupPS). Workers and servers can exist on a machine in a mutually exclusive fashion or workers can