Course Projects

1. BIG DATA DEMYSTIFIED

Infrastructure for supporting big data analytics are complex systems composed of many independent and intricate components. Over the course of the last few years, the community has worked effortlessly to improve, in isolation, each of these components. With advancements in the scheduling disciplines, in the network architecture, in the mechanisms for effectively sharing the network, in mechanism for enforcing low latency, and more recently in cross cutting techniques to improve several of these dimensions.

A particularly interesting question that arises given the phlethora of techniques is this: "what is the right combination of components for a given workload and a given environment?" This question is motivated by several different important factors; on one hand the high level analytics applications for different enterprises have radically different properties. It is ridiculous to expect techniques developed for the likes of Google and Facebook to prove effective when used by small business customers of Cloudera. Furthermore, while it is quite acceptable to believe that all companies, small and large, purchase similar commodity switches and relatively similar commodity servers. However the difference in scale, changes the failure properties. Failure will be discussed as part of project 2.

To start off the aim of the project is to understand the appropriate combination of software components that promotes a "predictable" and "safe" environment for running analytics jobs. An implicit assumption is that the right combination of components is dependent on the environment and the type of applications being run. The googles of the world have the luxury of warehouse scale computing and also of multi-terabyte applications. Where-as the multitudes of smaller enterprises, use much smaller clusters and have limited sized applications with limited data sets. Thus the question is, given an environment and a set of candidate applications, “is it possible to determine the set of combinations that promotes "safe" and "predictable" big data analytics?”.

The high-level goal of this project, is to examine combinations of the various components and determine an appropriate combination :

- Compute Scheduling: Quincy Scheduler, Delay Scheduling, Default Scheduler
- Network Sharing (Congestion Control): TCP, seawall, Orchester
- Network Topology: Tree, VL2, Fat-tree, Helios/C-through
- Compute Sharing: Mesos, Omega
- Performance Hacks: dynamic # of replicas (Scarlett), Cloning tasks(dolly), speculative execution (LATE, Maentri)

There are two ways to perform this analysis. The first is to use simulations and the second is to run on a testbed. We’ll focus on the simulator for now.

1.1 Simulators Evaluation

There are a couple of simulators; HRSim, MRPerf. The first task is to examine the veracity of the various simulators and understand the simplicity with which the various simulators can be modified. Of particular interests are veracity of the following:

- Scheduling Modules/Code
- Network Module/Code
- Speculative scheduling Code

1.2 Traces

To understand both ends of the spectrum the combinations will be evaluated with traces from Facebook and from Cloudera. The facebook traces will provide a means of understand large data whereas the Cloudera traces will focus on the small data. In essence both traces will provide complimentary portions of the whole spectrum.

2. FAILURE-AS-A-SERVICE

Hadoop, MapReduce, and Dryad are all built to leverage commodity switches to support embarrassingly parallel computing. A notably significant property of commodity devices is failure. However it is not immediately obvious how failure, independent and correlated, impact the set of jobs being run within the network. More so, it is not clear how failure impacts big data as a function of changes in cluster size and properties or a function of data size and application scale.

One approach is emulate google and replicate data to extreme lengths as a means of overcoming failures. Unfortunately, this option is not available to all companies. Motivated by this, there is a need to understand:

- how much replication is needed as a function of scale.
- is replication always successful strategy
- how does failure affect job completion times
- what are the implications of failure on outliers/straggers
- finally does failure have different implications when used with different combinations (combinations described in Project 1?

2.1 Fault Model

The first step, is generating a model for inject faults into a cluster. This model can build on the plethora of work done on understanding failures in Data center, in WAN, and in servers. The challenges lie in injecting correlated and independent faults. Similarly, in ensuring that the right distributions are being preserved as faults are injected.

3. CACHING IN BIG DATA REVISITED

Prior work have illuminated the ziphian property of input data popularity – an attribute that motivates the caching of input as a means of improving job completion times.

This input data is often fed to a map process that performs user defined processing. However given the rigidity of the map-reduce paradigm, a fundamental question arises, how diverse are the set of map functions being applied on the input data? Intuition suggests a limited number of such functions.

Given this assumption, then a naturally conclusion from this is that intermediate output may infact also display ziphian properties. Implying that caching intermediate output and tremendously improve the completion time of Map-Reduce jobs.

This project can be viewed as a two phase project. In the first phase, traces from Cloudera and possibly Facebook map-reduce will be examined, with an towards:

- First, confirming ziphian distribution of input data to map-reduce.
- Second, quantifying the number of different classes of map computation
- Third, understanding the distribution of intermediate output.
- Fourth, quantifying the popularity of intermediate output
- Fifth, developing a simulator to quantify the benefits
- Six, developing a system to realize these benefits

3.1 Understanding intermediate output

The traces do not present intermediate output in a manner that easily lends itself to the analysis we aim to perform. A trivial way to recast the intermediate output is to applying a hashing function or simple naming scheme.

The intermediate output should be renamed as following: New_name = input_data:map_function A hash can then be applied over this name to summarize or compress it Hash(New_name)

Using this name we can define and quantify the popularity of each intermediate data. We can answer the burning question, is intermediate output ziphian or does it follow a distribution that makes it amenable to caching. Further of particular importance is understand how long data can be cached for and eventually how much caching is required.

3.2 Benefits of Caching

Should the intermediate output follow a distribution that allows for caching, a natural follow on question then becomes what are the benefits of caching. Caching provides two high level benefits, it eliminates the need to rerun the map tasks thus freeing up computation nodes for other processes. Second and correlated, . Third, reducing the need

to transfers input blocks reduces the networking overhead of each mapreduce job. Finally, caching of this intermediate data can perhaps be used to reduce the shuffle phase.

A simple simulator should be developed to perform the following:

- Quantity savings from caching; reduction in Network transfers and reduction in CPU cycles used by a job.
- Quantify the amount of space required for caching and analyze how this changes overtime.
- Quantify the duration of popularity for each intermediate block.

4. BACON AND HONEY: DISSECTING PIG AND HIVE

An interesting and unexplored space lies in the understanding of user developed big data applications. Specially, understanding the high-level jobs. An interesting study [] had been performed to understand and quantify map-reduce jobs. The goal is to perform a similar study but at a higher level. The greatest challenges lies in developing a taxonomy of jobs and finding a way to automatically bin jobs into these different classes. Given a good enough classification system, then a very rich study can be performed. To start out with we can replicate the study performed by []. A very interesting study that is bound to generate excited within the community.

Even more enticing would be to understand the differences in how the various classes within this taxonomy are translated into map-reduce jobs.

This study will be backed by traces from Cloudera. OUr findings will definitely be used to influence operations at Cloudera, even more so the linkage between the high level Hive/Pig and map reduce can eb explored by recreating results and perhaps interestingly enough by even changed. we can try to understand way to perform the transformations.

5. CLOUD PERFORMANCE DEMYSTIFIED

Performance-related issues are especially challenging to diagnose in the cloud. First, there can be several possible causes of the performance degradation and accurately identifying the root cause is difficult. Equally difficult is pinpointing the scope of a problem's impact – e.g., whether it affects just the tenant, or all tenants sharing a server, a rack, an entire data center or zone. Second, the high degree of multiplexing means that performance (e.g., of compute and I/O resources) observed by a tenant is quite variable and is dependant on many factors, such as the level of oversubscription and workload patterns of other tenants in the cloud. This makes traditional approaches to problem diagnosis, such as those based on deviations from an expected baseline, inapplicable. Thus, it is no surprise that root-cause investigation of performance issues is often the lengthiest step in the cloud problem management process, consuming significant operator attention both from the cloud provider and the tenant [**?**].

In this project, we aim to design a framework that helps tenants of large shared IaaS clouds to quickly and accurately distinguish the likely cause and scope of performance issues they are experiencing. The framework hinges on the intuition that performance problems occurring in the cloud provider's infrastructure are likely to affect virtual instances belonging to multiple tenants; the number of tenants observing similar issues depends on the scope of the problem's impact. By enabling tenants to share information and helping correlate across them, the framework facilitates effective problem diagnosis and response. Thus, framework is collaboration-based and tenant participation-driven.

There are several challenges to realizing this framework:

First, performance within the cloud is fundamentally unpredictable. This variation is significant for micro-instances and for small instances. However, this becomes less so with medium and large instances. Thus the mechanism must extract predictability and infer features that imply predictability.

Second, even in the face of predictability differences in the I/O performance of different tenants is impacted by the type of operating system, file-system, in use. Thus these differences must be normalized or eliminated.

Towards achieving this framework, a few steps must be taken:

- Quantify OS/File System differences: Download all the images from EC2 and then try to understand the distribution of Operating System and File systems in use.
- For each instance type-zone combination develop a distribution of observed performance
- Quantify performance variation in all types of instances (micro–large).
- Quantify differences between zones.

The answer to the above questions will help dictate how the framework should be developed going forward.

6. UNDERSTANDING END-USES

Invariable big data analytics is geared towards providing services consumed by users. Of particularly importance then is the efficiently of the delivery system between data center hosting big data architectures and the devices used by the end users. Prior studies have shown that latency significantly matters and driven by this Google and other have sought various ways to improve TCP by reducing the impact of the TCP handshake, by introducing great levels of pipelining or parallelism, and my improving the robustness of TCP to loss.

A particularly important point, is that many of these mechanisms violate the fairness properties of TCP, more specifically components proposed in. Even more, it is not clear how the mutated versions of TCP interact in the wild or how they are impacted by the multitudes of middleboxes and switches. Thus a high-level goal is to perform a measurement study aimed at understanding to a larger extent the properties of traffic as viewed by the end-user.

There are several high-level questions to understand:

First, recent work at Sigcomm 2013 have shown that in mobile networks large buffers affect the performance of end-users. Broadband have similarly large buffers however little has been done to examine and discover similar properties. The goal is to understand the impact of large buffers on user latency.

Second, assuming that TCP connections can be finger-printed by how they interprete events, how many such finger-prints are there in the wild? How does these finger-prints differ? What are the implications for fairness.

Third, given that the internet's ecosystem is dominated by a few entities. It a particuarly interesting to understand how interactions with these entities impact the interface. For each provider (Google ... Facebook), can we quantify the amount of traffic to each, the high level properties of these flows (rate, duration, size), the low level properties (finger-print, TCP type), and the interaction patterns (dependency graph of connections).

6.1 data-set

- **Bismark:** Nick Feamster at GTech has offered to give access to traces from the Bismark routers. Bismark routers have been deployed in a few 100 homes as gateway devices and they capture packet traces from end users.
- **U of Wisconsin:** The Wisconsin IT department is will to provide us with traces taken from users of its wireless networks. This could provide traces from 10000 of users.

7. HADOOP-AS-A-SERVICE DEMYSTIFIED

There are a couple of dominant form of IaaS clouds, ones provided by Amazon, Azure, and Google. Several have started to offer Elastic Hadoop or hadoop as a cloud service. Of particular importance is trying to reserve engineering this service and trying to understand how it is deployed.

A few interesting things to understand:

- How is storage handled? EBS? Local Storage? HDFS?
- What are the network guarantees of the Cluster?
- What is the topology?
- is Hadoop run natively or is there a layer of virtualization?

In general, very little is understood about how the cloud providers provide the services that they provide. Any attempt to shed light on this is received positivity.