Panthera: A Study of Caching in Distributed Computing

Background

Distributed computing has seen tremendous growth in the past few years. Essentially, **it allows computational algorithms to run across multiple computers (i.e. nodes)**. Most distributed systems must make files accessible over networks of computers. **This project aims to reduce the waiting time (known as latency) associated with obtaining files within distributed computing systems**.

Introduction

Hadoop, an open source project that allows developers to write distributed applications, has found extensive use in academia and industry. For example, Yahoo, Facebook, Stanford Computer Science, etc. are all heavily involved with Hadoop development. But, Hadoop does not effectively utilize Random Access Memory (RAM) and local storage to better performance. **This project develops and tests scheduling and caching mechanisms to reduce waiting time (i.e. latency) in Hadoop, thereby greatly increasing Hadoop's efficiency and applicability in fields ranging from medical diagnosis to artificial intelligence.**

Caching

There is significant waiting time/latency associated with retrieving files from distributed filesystems (Griffioen 1994). **Caches** are used to store some files locally so that they can be accessed much more quickly. Panthera implements a cache and a cache-based scheduler for Hadoop.

Problem

- **Develop a caching layer** for the Hadoop Distributed File System that can cache **both data and metadata**.
- The solution should be a **"drop-in" addition**, allowing for easy adoption and integration into existing systems.
- The layer **should reduce file access latency**/waiting time significantly in comparison to **the control group: a standard Hadoop installation**.
- **Develop a scheduler** for the Hadoop which **schedules jobs based on what is available in the cache**. In addition, it should be be able to **prefetch files that could be needed in the future**.

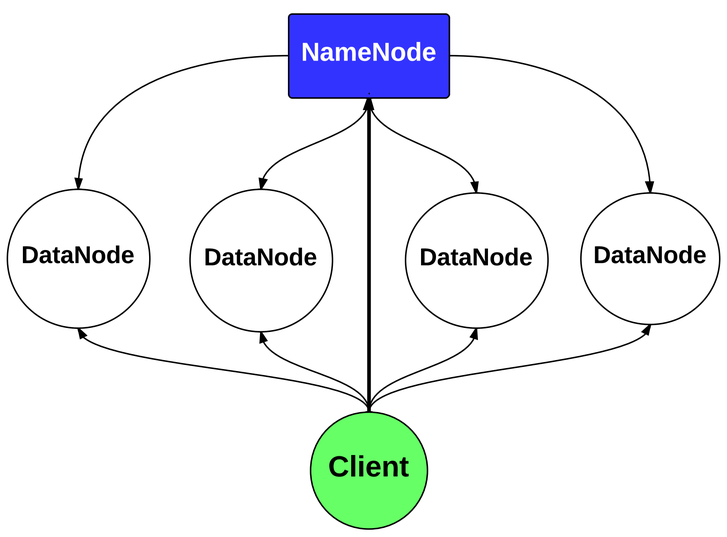
Hypothesis

The cache layer designed (called **Panthera**) will have lower file access latency when compared to a standard Hadoop installation. Additionally, it will be possible to design **Panthera** such that the Hadoop source code remains unaffected. It will also be feasible to create a cache-based scheduler for Hadoop.

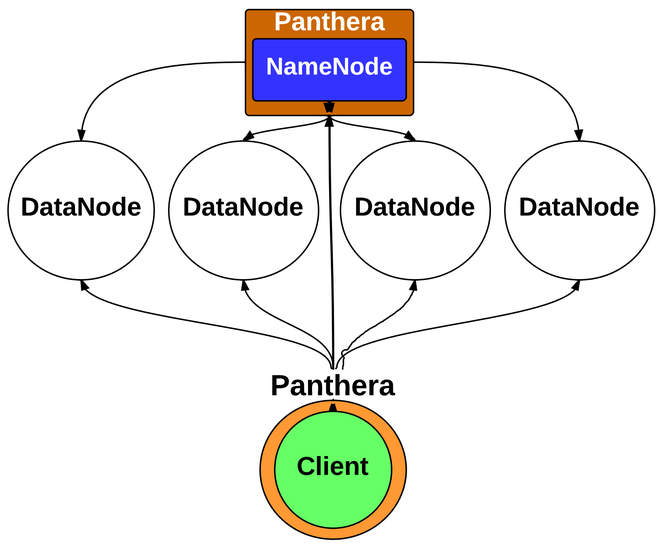
Data vs Metadata

Data within the Hadoop file system **consists of the actual contents of a file. Metadata is the information about files and directories**, e.g. time a file was created, the list of files in a directory, etc. **Panthera performs both metadata and data caching; both have significantly different technical implications and challenges.**

Current Hadoop File System Architecture



Panthera Architecture



Panthera - Implementation

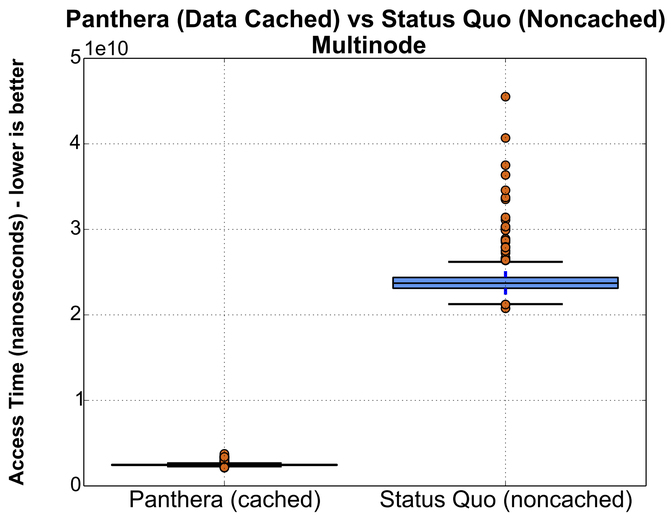
- **Panthera is implemented on Google's Go programming language** which has excellent concurrency primitives and memory management, making it perfect for for software like **Panthera** that must run reliably with hundreds of clients.

Latency Testing Methodology

Client and server (i.e. NameNode or DataNode) are on separate machines for both metadata and data cache testing.

- **Metadata**: Repeatedly request directory listing for a directory with 100 files in it.
- **Data**: Repeatedly request file contents for a 1MB large file.

Latency Results

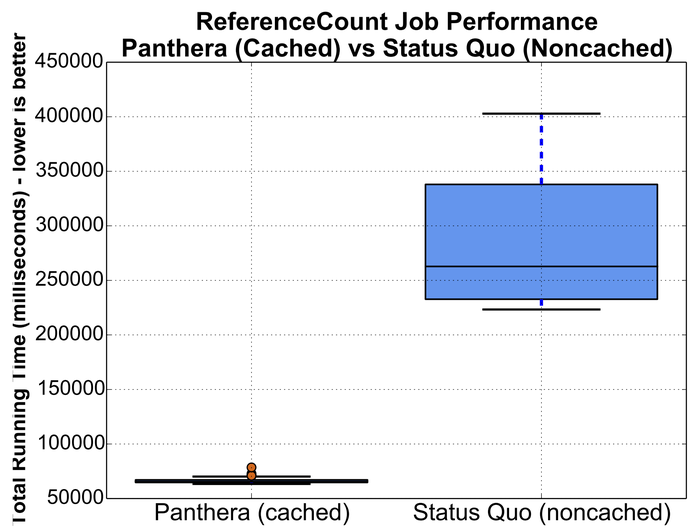


Procedure Testing Methodology

Panthera was also tested with running times of algorithms running on Hadoop (as opposed to simply downloading files). The input sample was a repeated copyof "The Adventures of Sherlock Holmes" (192 MB).

- **WordCount**: The classic distributed algorithm that counts occurrences of all words in a given text.
- **ReferenceCount**: Counts occurrences of words in a given reference file (i.e. dictionary).

Procedure Results



Latency Types

In the Hadoop File System Architecture, there are several types of latency involved (listed from highest to lowest latency):

- **Network latency**
- **Hard drive latency**
- **Memory latency:** Reading from RAM memory has an extremely small waiting time associated with it. **Panthera converts hard drive latency to memory latency, thereby greatly reducing file access time.**

Statistics

Type	Mean time improvement
Metadata	862.6%
Data	863.4%
Word count	25.1%
Ref. count	331.6%

Table: Improvement with Panthera in the total amount of time required to execute algorithms/requirements (compared with status quo)

Type	Standard deviation contraction
Metadata	27.38x
Data	11.14x
Word count	0.78x
Ref. count	22.91x

Table: Reduction obtained with Panthera in the standard deviation of running times, i.e. a contraction of 9x implies that the standard deviation was reduced by a factor of 9 with Panthera (compared to the status quo).

Conclusion

The results validate the hypothesis presented.

- Panthera was successfully developed as a "drop-in" solution
- It was able to reduce latency in comparison to a standard Hadoop installation.
- A scheduler was also implemented which uses cache information to schedule Hadoop jobs.
- Metadata latency was **decreased by a factor of 7.36**
- Data latency was **decreased by a factor of 7.634**
- The running of time of WordCount (an algorithm with very few file accesses) was **improved by 25%** with the code left unmodified.
- The running time of RefCount was **improved by 331.6%**.

As a "drop-in" solution, Panthera can be employed in a variety of applications to improve performance without code modification. With the demonstrated time and resource savings, Panthera can have incredible impacts in distributed systems and in specific applications.

Applications

Hadoop has diverse applications and since Panthera was created as a "drop-in" solution, the **applications span an extremely wide range of disciplines**.

- Bioinformatics (e.g. CrossBow from Johns Hopkins University)
- Numerical/scientific computing
- Image processing
- Artificial Intelligence/Machine Learning (e.g. Apache Mahout)
- Market Research (e.g. Target)

Future Work

- **All code related to this project will be open sourced** to further development in distributed computing.
- Currently working on testing a **bioinformatics toolkit with Panthera**; initial results are very promising and show significant reduction in total running time.
- Smart scheduling using Panthera; arranges tasks so that the cache is used most efficiently.