Pfimbi: Accelerating Big Data Jobs Through Flow-Controlled Data Replication

Simbarashe Dzinamarira*
Florin Dinu△ T. S. Eugene Ng*

*Rice University, △EPFL
DFSs have a critical role on the Big-Data landscape

- Rich ecosystem of distributed systems around Hadoop and Spark
- Predominantly use HDFS for persistent storage
- A performant HDFS benefits all these system

Synchronous data replication in HDFS and its shortcomings

- Bottlenecks affect the whole pipeline
- Contention between primary writes and replication
Synchronous replication seldom helps boost application performance

- In a study by Fetterly et al. only about 2% of data was read within 5 minutes of being written [TidyFS: USENIX ATC 2011]
- Fast networks reduce the cost of non-local reads
- There can be data locality without replication
Synchronous replication impedes industry efforts to improve HDFS

- Heterogeneous storage

- Memory as a storage medium

SSD image from: [http://www.storagereview.com/intel_ssd_525_msata_review](http://www.storagereview.com/intel_ssd_525_msata_review)
Asynchronous replication relieves the effects of pipeline bottlenecks.
Beside asynchronous replication, we need flow control to manage contention.
Pfimbi effectively supports flow controlled asynchronous replication

- Allows diverse flow control policies
- Cleanly separates mechanisms from policies
- Isolates primary writes from replication
- Avoids IO underutilization
Pfimbi Overview

• Inter-node flow control

DATANODE

DATANODE

• Intra-node flow control

DATANODE

SSD image from http://www.storagereview.com/intel_ssd_525_msata_review; Magnifier image from https://commons.wikimedia.org/wiki/File:Magnifying_glass_icon.svg
Inter-node flow control

- Client API: (# of replicas, # of synchronous replicas)
- Timely transfer of replicas to ensure high utilization
- Flexible policies for sharing bandwidth

![Diagram showing inter-node flow control with areas labeled for synchronous and asynchronous operations, and a client API notation.](image-url)
Hierarchical flow control allows Pfimbi to implement many IO policies

- Example 1: prioritize replicas earlier in the pipeline
- Example 2: fair sharing of bandwidth between jobs
Intra-node flow control

- Isolate synchronous data from asynchronous data
- Avoid IO underutilization
Intra-node flow control

Pfimbi’s strategy

- Keep the disk fully utilized
- Limit the amount of replication data in the buffer cache

Threshold for asynchronous replication: \( T + \delta \)

OS threshold for flushing buffered data: \( T \)

**Typical Values**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( T )</td>
<td>10% of RAM (~13GB)</td>
</tr>
<tr>
<td>( \delta )</td>
<td>500MB</td>
</tr>
<tr>
<td>Buffer Cache</td>
<td>20% of RAM (~26GB)</td>
</tr>
</tbody>
</table>
Additional topics that are discussed in detail in the paper

- Other activity metrics and their shortcomings
- Consistency
  - We maintain read and write consistency
- Failure handling
  - Same mechanism as in HDFS to recover from failures
- Scalability
  - Pfimbi’s flow control is distributed
Evaluation

• 30 worker nodes
  • NodeManagers collocated with DataNodes

• 1 Master node
  • ResourceManager collocated with NameNode

• Storage
  • 2TB HDD
  • 200GB SSD
  • 128GB DRAM
Pfimbi improves job runtime and exploits SSDs well

**DFSIO on HDFS**

<table>
<thead>
<tr>
<th>HDD-&gt;HDD-&gt;HDD</th>
<th>SSD-&gt;HDD-&gt;HDD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Completion time of replicas (s)</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>100</td>
</tr>
</tbody>
</table>

**DFSIO on PFIMBI**

<table>
<thead>
<tr>
<th>HDD-&gt;HDD-&gt;HDD</th>
<th>SSD-&gt;HDD-&gt;HDD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Completion time of replicas (s)</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>100</td>
</tr>
</tbody>
</table>

- **Primary write**
- **1st replica**
- **2nd replica**
- **Syncing dirty data**
Necessity of flow control when doing asynchronous replication

Two DFSIO jobs

Completion time (s)

Without Flow Control

With Flow Control

Configurations

Job 1
Job 2
Remaining replication
Pfimbi performs well for a mix of different jobs: SWIM workload

18% IMPROVEMENT IN AVERAGE JOB RUNTIME
Policy Example: Pfimbi can flexibly divide bandwidth between replica positions

**Equal weights**

**Weights in ratio 100:10:1**
Related Work

• Sinbad [SIGCOMM 2013]
  • Flexible endpoint to reduce network congestion
  • Does not eliminate contention within nodes

• TidyFS [USENIX ATC 2011]
  • Asynchronous replication
  • No flow control leads to arbitrary contention

• Retro [NSDI 2015]
  • Fairness and prioritization using rate control
  • Synchronous replication
Conclusion

• Pfimbi effectively supports flow controlled asynchronous replication

  • Successfully balances managing contention and maintaining high utilization

  • Expressive and backward compatible with HDFS