Understanding Issue Correlations: A Case Study of the Hadoop System

Jian Huang  Xuechen Zhang*  Karsten Schwan

Georgia Institute of Technology  * Washington State University Vancouver

Motivation

➢ Are there correlations between issues in the Hadoop system?
➢ Which types of issues appeared most frequently in MapReduce and HDFS subsystems, respectively?
➢ What is the correlation between root causes of these issues and characteristics of the subsystems?
➢ What are the consequences, impact, and reactions of the issues?

Methodology

➢ Target issues: 2180 HDFS and 2038 MapReduce issues reported between 10/21/08 and 08/15/2014.
➢ Our focuses: commit time, type, priority, causes, consequence, impact, correlated issues.
➢ Approach: issues are examined by two observers separately, and discussed until consensus was reached.

Issue Overview

➢ Results related to issue types and priority
  ✓ Bugs dominate the solved issues.
  ✓ Minor issues can significantly affect system availability and serviceability, and some of them are not easily fixed.
  ✓ Similar issue patterns are observed over time for both HDFS and MapReduce.

Issue Correlation

➢ Key findings
  ✓ Most issues are independent.
  ✓ HDFS issues tend to relate to issues in Hadoop Common (62.5%), Hbase (15.0%), and YARN (10.0%).
  ✓ MapReduce issues tend to relate to issues in YARN (46.3%), Hadoop Common (29.0%), and HDFS (9.0%).
  ✓ 26% of HDFS and 33% of MapReduce issues have similar causes.
  ✓ Correlated issues require almost twice the fix time of independent issues.

Issue Consequences and Impact

➢ Consequences
  - Deadlocks, inconsistency, out of memory, non-existent objects
  - Wrong block operations and data layout changes
  - Inappropriate usage of exceptions and bugs in fault handlers
  - Execution in unexpected path and output issues

➢ Common causes
  - System reliability is the most vulnerable aspect in Hadoop; many availability issues were triggered in fault handling methods.

Reaction to Issues

➢ Exceptions: widely used to catch error signals; exception handling itself is error-prone.
➢ Retrying: overcome transient errors; it can result in system hangs or failures.
➢ Silent reactions: handle minor issues; it can cause severe problems like data loss and service unavailability.
➢ Recovery: 3.5% of the issues relate to recovery with checkpointing.

Related Work

➢ Bug and patch analysis in various systems
  ✓ Cloudera’s CDH3 Hadoop distribution
  ✓ 3655 ‘major’ issues in cloud systems
  ✓ Conventional Linux file systems and Linux kernels

Similar motivations: to learn from mistakes and experience, our unique focus: to reveal the issue correlations with characteristics of distributed systems.

➢ Results from existing bug-finding tools
  ✓ Many failures are caused by error handling, e.g., fault handler is not implemented.
  ✓ Use the logs to reproduce failures.

Our observations: (1) many issues are caused by inappropriate usage of exceptions and by incorrect logic in fault handler implementation; (2) logs should be audited to reduce false positives.

Conclusion

➢ Most of the Hadoop issues do not depend on external factors.
➢ Half of the issues are internally correlated, such as those occur for fixing other issues, or block fixing other issues.
➢ The root causes of the issues have strong correlations with the subsystem characteristics.
➢ Our study offers useful hints and findings to assist in the development of bug-finding tools.