**密级:**



**硕士学位论文**

**大数据系统应用可靠性测试框架设计与实现**

**作者姓名： 郑莹莹**

**指导教师: 叶丹 研究员**

**中国科学院大学软件研究所**

**学位类别: 工学硕士**

**学科专业: 软件工程**

**培养单位: 中国科学院大学软件研究所**

**2017年 4月**

**Design and Implementation of a reliability benchmarking framework for big data system**

**By**

**Zheng Yingying**

**A Dissertation Submitted to**

**University of Chinese Academy of Sciences**

**In partial fulfillment of the requirement**

**For the degree of**

**Master of Computer Software and Theory**

**Institute of Software**

**University of Chinese Academy of Sciences**

**April, 2017**

**独创性声明**

本人声明所呈交的论文是我个人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明。

签名：\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 日期：\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**关于论文使用授权的说明**

本人完全了解中国科学院软件研究所有关保留、使用学位论文的规定，即：中国科学院软件研究所有权保留送交论文的复印件，允许论文被查阅和借阅；中国科学院软件研究所可以公布论文的全部或部分内容，可以采用影印、缩印或其它复制手段保存论文。

（保密的论文在解密后应遵守此规定）

签名：\_\_\_\_\_\_\_\_\_\_ 导师签名：\_\_\_\_\_\_\_\_\_\_ 日期：\_\_\_\_\_\_\_\_\_\_

大数据系统应用可靠性测试框架设计与实现

摘要

当前，Spark、Flink等分布式处理框架被广泛应用于大数据的处理分析，然而在处理大数据时，经常出现内存溢出、IO异常、任务超时等运行时错误，这些错误会直接造成应用执行失败。然而，现有的测试基准主要针对系统的性能提供测试，没有考虑系统的可靠性问题；并且，这些测试基准通常使用常规数据以及固定的配置来进行测试，缺乏测试的多样性。针对大数据系统面临的多样及复杂的运行时错误以及现有测试基准的不足，论文设计并实现了面向大数据系统应用的可靠性测试基准框架，提供可视化的测试界面，支持自定义的数据生成和自动参数组合测试。

设计与实现大数据系统应用的可靠性测试基准框架，论文介绍了可靠性测试基准的设计，并针对以下关键技术进行研究。首先，本文提出了针对大数据应用的异常数据生成方法，定义了异常数据的概念和数据的多种随机分布形式；同时，通过分析应用程序特征，提出应用计算特性与数据异常特征对应关系，并给出了特定应用的异常数据的生成规则。其次，本文提出了针对大数据应用的参数组合测试方法，采用组合测试，并通过贪心算法对系统和应用参数进行组合空间削减测试；针对不满足参数独立性以及参数取值的相关性的，提出探测性参数验证方法，通过指数增长的慢启动方式来确定最差的资源占用的参数取值。

论文详细的介绍了大数据系统应用可靠性测试基准的设计与实现，并将该可靠性测试基准框架应用在Spark系统上。通过对Spark上的基准应用进行可靠性测试，验证了本文提出的异常数据生成以及参数组合测试方法的可用性，并在6个应用（Join、Mix、RandomForest、LogisticsRegression、ALS、PageRank）中发现了三种类型的错误（内存溢出错误、运行超时以及计算结果错误）。

**关键词：**可靠性，基准测试，异常，大数据系统，大数据应用

**Design and Implementation of a reliability benchmarking framework for big data system**

**ABSTRACT**

**KEY WORDS:** reliability, benchmark, error, large data system, large data application

目录

[第一章. 绪论 1](#_Toc478549824)

[1.1 研究背景 1](#_Toc478549825)

[1.2 研究内容 2](#_Toc478549826)

[1.3 论文组织 3](#_Toc478549827)

[第二章. 大数据系统应用可靠性相关工作 5](#_Toc478549828)

[2.1 大数据系统及应用 5](#_Toc478549829)

[2.1.1 大数据系统 5](#_Toc478549830)

[2.1.2 大数据应用 8](#_Toc478549831)

[2.2 大数据系统及应用的可靠性问题 9](#_Toc478549832)

[2.2.1 可靠性定义 9](#_Toc478549833)

[2.2.2 可靠性问题分析 10](#_Toc478549834)

[2.3 测试基准框架研究现状 11](#_Toc478549835)

[2.4 测试基准需求 13](#_Toc478549836)

# 绪论

本章介绍论文的研究背景、主要工作和论文组织方式。

## 研究背景

随着互联网、移动互联网、物联网的发展，越来越多的领域产生了“海量”和“高速”的数据[1]。例如，在金融领域，其日常运营过程中会产生大量的数据，这些数据产生速度快，并且时效性短[2]。又如，在移动通信领域，一个大型城市中每分钟都会有超过8万条的位置更新数据，每天的网络承载流量高达100TB[1]。这些数据具有数据规模大，产生速度快的特点，而且在这些数据中也隐藏着巨大的价值。在金融领域，通过对这些海量数据进行计算和分析，发现隐藏在其中的特征，可以帮助金融行业进行实时决策，从而更好的进行风险管理以及实现商业智能化。在移动通信领域，如果可以对这些海量数据进行实时的挖掘分析，可以降低电话诈骗造成的损失。因此，对大数据应用场景下的海量数据的处理分析已经成为一个迫切的需求。

为了应对海量数据以及发现其背后隐藏的巨大价值，许多大数据系统及大数据应用应运而生。常见的大数据系统有Apache Storm[4]、Apache Hadoop[5]、Apache Spark[6]、Apache Flink[7]等。这些分布式处理框架被广泛应用于社交网络、搜索引擎、数据采集及数据查询等应用场景中，而这些应用场景又衍生出了一系列用于处理特定领域的大数据应用。常见的大数据应用有SQL查询、大规模图分析、机器学习以及流式应用等。然而，现有的大数据应用在处理数据时，经常出现内存溢出、I/O异常、任务超时等运行时错误，以及在流处理过程中的数据完整性和计算结果错误等。这些可靠性的问题将会直接造成应用执行失败，甚至产生更加严重的后果。

通过开源论坛、社区、bug issues以及相关论文的研究发现，本文分别分析了产生运行时错误以及计算完整性问题的原因。（1）首先，对于I/O异常、内存溢出及任务超时等运行时错误，其产生原因[8][9][10]包括不恰当的配置参数（输入的数据块过大，分区数目过小，分区函数不均衡等）、异常数据（数据维度过高、数据倾斜等）以及用户代码缺陷（内存泄漏、较高的时间或空间复杂度等）等。（2）其次，对于数据错误（如数据丢失、数据重复等）以及状态错误（状态丢失、快照错误等）等数据和计算完整性问题，根据博客、论坛等的实证分析[11][12][13]，发现其产生原因主要有数据流速过快、Task失效以及快照恢复机制不完善等。在上述分析的基础上，本文将运行时错误以及计算结果错误等问题产生的原因总结为以下三点：（1）系统错误，包括硬件错误（如CPU、内存、网络以及硬盘等错误）和软件错误（如设计缺陷、逻辑错误、实现bugs等）。（2）应用错误，如参数配置不当或代码缺陷等。（3）数据异常，如数据维度过高或数据倾斜等。

对于用户来说，他们希望了解部署好的应用是否存在潜在的应用错误；对于系统设计者和管理者来说，他们希望了解部署的或更新的系统是否有潜在的系统错误。目前，针对这些应用错误和系统错误的解决方案[14][15]通常是在出现问题之后，再针对某一类应用及问题进行分析诊断。通过测试来发现错误是一种常用的方法，然而当前还没有一个通用的检测方法能够提前发现系统、应用和数据存在的潜在问题。

通过调研发现，现有的大数据平台测试基准，如HiBench[16]、BigSQL benchmark[17]、Spark-perf[18]、Graphalytics[19]以及SparkBench[20]等，关注的焦点通常是特定的大数据平台上的性能或扩展性的基准测试；并且，主要使用常规的输入数据（固定的真实数据集或其简单合成）以及固定的参数配置进行测试。这些测试基准都不能直接用于检测潜在错误，因而对于大数据系统的可靠性测试，目前还没有一套类似的测试基准提供支持。

大数据系统在处理“海量”和“高速”数据时，是否可以高可靠的应对高负载的场景已经成为一个亟待研究的课题。针对大数据系统面临的可靠性问题以及现有测试基准的不足，本文将研究如何开发一个针对大数据系统应用的可靠性测试基准框架。该框架的系统目标主要包括以下几点：

1. 构建一个大数据系统的可靠性测试基准，选取SQL查询、大规模图分析以及机器学习应用领域中使用广泛的典型应用作为基准应用；提供包含异常数据的丰富多样的测试数据；提供支持参数组合的测试。
2. 集成数据生成、组合测试以及测试报告生成模块，构建一站式可靠性测试基准平台，将测试人员从测试脚本编写以及手动执行测试等繁琐工作中抽离出来。
3. 提供基于Web的可视化界面，可配置的数据生成以及参数组合测试界面，降低测试人员的使用难度，加快测试速度；自动生成的测试报告，降低分析人员的分析成本。

## 研究内容

针对大数据系统在特定的应用领域面临的运行时错误等可靠性问题，本文主要研究如何通过提前测试的方法来发现系统、应用和数据存在的潜在问题。根据现有测试基准存在的不足，本文将研究如何设计并实现一个大数据系统应用可靠性测试基准框架。

首先，可靠性测试基准需要选取SQL查询、大规模图分析以及机器学习等应用领域中使用广泛并且具有计算特征的典型应用作为基准应用。其次，针对这些基准应用，可靠性测试基准需要提供测试数据；为了检测潜在的应用错误或系统错误，测试数据除了提供常规数据，还应根据应用计算特性生成符合异常数据特征的异常输入数据。最后，在使用测试数据对基准应用进行测试过程中，可靠性测试基准需要提供参数组合测试来检测极端配置参数可能会造成的错误；考虑到组合测试中参数组合空间过大的问题，可靠性测试基准需要基于参数特征减少参数组合空间。

根据上述研究内容，该大数据系统可靠性测试基准框架开发中的关键技术，主要包括以下几点：

1. 提出了针对大数据系统应用的数据生成方法，提供异常数据的生成。首先定义了异常数据的概念，并通过分析应用程序特征，给出了特定应用的异常数据的生成规则，进而生成相应的异常数据（如分布异常、维度过高等特征）。
2. 提出了一种针对大数据系统应用的参数组合测试方法。通过参数组合，并应用贪心算法对系统和应用参数进行组合空间削减。同时，通过指数增长的慢启动方式对参数相关性进行探测性验证，从而确定最差的资源占用的参数组合情况。

最后，论文介绍了大数据系统的可靠性测试基准框架的实现，并将该可靠性测试基准框架应用在Spark系统上。通过对Spark系统中的基准应用进行可靠性测试，已经在6个应用中发现了三种类型的错误（如，内存溢出错误、运行超时以及计算结果错误）。同时，通过这些错误验证了可靠性测试基准的异常数据生成方法以及组合参数测试方法的可用性。

## 论文组织

论文的后续章节组织方式如下：

第二章介绍大数据系统及应用可靠性相关工作的研究。首先，介绍了大数据系统和大数据应用的研究现状；接着，介绍了大数据系统及应用面临的可靠性问题，包括可靠性定义以及可靠性问题的分析；最后，介绍了测试基准框架的研究现状。

第三章介绍可靠性测试基准的设计。分别介绍了基准需求，基准应用，测试数据，基准的执行以及测试度量。

第四章介绍可靠性测试基准的关键技术。首先，定义了异常数据概念以及数据概率分布形式，并介绍了数据生成方法；然后，介绍了参数配置，组合测试以及参数组合空间削减策略。

第五章介绍大数据应用可靠性测试基准框架实现。首先，介绍了系统架构；然后，介绍了系统实现，包括系统总体设计、数据生成器以及组合参数发生器；最后，对实现的可靠性测试基准框架进行应用验证，介绍了实验环境、参数配置和可靠性测试的实例及分析。

第六章总结了全文，主要包括论文的工作、论文贡献以及对未来工作的展望。

# 大数据系统应用可靠性相关工作

本章介绍了大数据系统及应用的发展现状，以及大数据系统应用面临的可靠性问题，最后介绍了测试基准框架的研究现状。

## 大数据系统及应用

随着社交网络、移动互联网、电子商务等技术的不断发展，以及云计算、物联网、互联网+等新兴技术的兴起，全球数据量正在急剧增长，大数据已经成为了当今时代的主旋律。这些网络通信、电子设备以及个人消费等产生的大数据信息，对于企业进行风险管理、分析用户需求以及政府管理交通治安等，都有着极大的帮助。为了处理和分析这些大数据，同时挖掘其背后隐藏的巨大价值，许多大数据系统和大数据应用迅速的发展起来。

### 大数据系统

目前，大数据的处理模式主要包括对静态数据的批量处理以及在线数据的实时处理[21]。因此，现有的大数据处理系统主要分为了批处理系统、流处理系统以及混合处理系统。

* 批处理系统

当前比较流行的批处理系统有Apache Hadoop[5]等。Hadoop是一种以MapReduce[3]作为处理引擎、以HDFS作为存储的批处理框架。MapReduce是一种分布式的编程模型，用于处理和生成大型数据集，并通过共享大规模系统集群，提供良好的可伸缩性。MapReduce将分布式编程分为了map和reduce两个阶段：（1）map阶段，用户通过指定*map()*函数来处理键/值对，以生成一组作为中间结果的key/value对；（2）reduce阶段，通过*reduce()*函数，将中间结果key值相同的key/value对合并。HDFS是一种提供可扩展的、可靠的、用于数据存储的分布式文件系统，通过使用大量低配置、低成本的服务器代替高配置、高成本的大型单机服务器，以及通过key/value对代替关系表[22]，为MapReduce处理产生的大型数据集提供了可靠的存储策略。

* 流处理系统

当前使用比较广泛的流处理系统有Apache Storm[23]等。Storm具有高度可扩展性，易于使用，并提供低延迟、有保证的数据处理，因此被广泛的应用于数据的实时处理中。Storm支持开发人员以任何编程语言实现他们的逻辑，同时还具有以下特点：（1）编程模型简单，Storm提供了一个非常简单的、类似于MapReduce的操作方式来提供实时处理。（2）快速可靠的消息处理，Storm提供“至少一次”的语义保障，一旦任务失败，可以从数据源恢复数据。（3）良好的扩展性，Storm提供多线程并发的进行流式计算，有良好的水平扩展的能力。

* 混合处理系统

当前应用较广的混合处理系统有Apache Spark[24]、Apache Flink[7]等。Spark和Flink同时提供了批处理和流处理两种大数据处理模式，并提供了更加丰富的生态系统，如图2-1和2-2所示。

Spark是一个基于内存的可扩展的大数据处理系统，它引入了一个称为弹性分布式数据集（RDDs）的抽象概念。Spark利用内存进行快速的数据计算，同时具有快速查询的优势，弥补了MapReduce中网络延迟及磁盘I/O过重的劣势[25]，这使得Spark在机器学习等应用中要比Hadoop快10到100倍。同时Spark Streaming可以亚秒级增量的对流进行缓冲，然后将这些缓冲作为小规模的固定数据集进行批处理[26]。这种使用微批处理的方式来处理流式数据的缓冲机制，提高了系统的吞吐率，但同时也带来了较高的延迟。因此，Spark Streaming并不适合处理对延迟要求过高的流处理场景。



图表 2-1 Spark生态系统

Flink也是基于内存的大数据处理框架，与Spark不同的是，Flink是一种支持批处理任务的流式数据处理框架。Flink基于同一个Flink运行时（Flink Runtime），提供流处理和批处理两种应用功能的支持，分别使用DataStream API和DataSet API。Flink提供了自己的内存管理，而不是完全依赖于JVM的内存管理机制，可以灵活的避免频繁GC（垃圾回收）带来的性能波动。Flink通过构建的DAG（有向无环图）来执行任务，其提供的Logical Plan可以实现自动优化迭代，同时在数据流运行时，实现了高吞吐速率和低延迟。



图表 2-2 Flink生态系统

图2-3和2-4分别展示了Flink和Spark的计算迭代流程。Flink提出了增量迭代的计算模型，在迭代中可以显著的减少计算。因此，Flink比Spark的流处理速度更快，内存利用率更高，更加适合流式计算以及迭代计算。



图表 2-3 Flink计算迭代流程



图表 2-4 Spark计算迭代流程

针对上述介绍的大数据系统，本文针对大数据系统的处理类型、计算模式、处理延迟以及处理速度等进行了总结，如表2-1。

表格 2-1 大数据系统对比

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **大数据系统** | **处理类型** | **计算模式** | **处理延迟** | **处理速度** | **成熟度** |
| **Hadoop** | 批处理 | 基于磁盘 | 高 | 慢 | 强 |
| **Storm** | 流处理 | 基于磁盘 | 低 | 快 | 强 |
| **Spark** | 批/流处理 | 基于内存 | 中 | 快 | 强 |
| **Flink** | 批/流处理 | 基于内存 | 低 | 快 | 中 |

在上述大数据系统中，Spark和Flink不仅提供了丰富的处理类型，具有更加丰富的生态系统，同时针对机器学习、交互式查询、大规模图计算等应用场景提供了良好的支持。Spark发展迅速，社区活跃，并广泛应用在了工业界；Flink以其高效的内存管理和增量迭代，在流式计算中有着很强的优势，并且也在飞速的发展着。对于被广泛应用于工业界的大数据系统，其可靠性更应引起广泛的关注。

### 大数据应用

互联网、云计算、物联网等应用场景中都会产生大量数据。这些应用场景主要包括社交网络、搜索引擎、数据采集及数据查询等，并根据这些应用场景衍生出了一系列用于处理特定领域的应用类别，如SQL查询、大规模图计算以及机器学习等。

* SQL查询

SQL是一种比较持久的查询语言，目前已经有广泛的工具支持它。基本的SQL查询类型主要包括原子查询、中间结果查询以及复杂查询。另外，Big Data benchmark，TPC-H以及TPC-DS等主流的数据库评测基准提供了更加丰富的SQL处理语句。

目前的大数据处理系统中，Spark提供的Spark SQL以及Flink提供的Table API和SQL都支持SQL查询的应用场景。

* 大规模图分析

当前的社交网络发展越来越迅速，Facebook、微博等建立了大量的社会关系网络，并且可以用图来表示人与人之间相互通信的关系。社交网络中复杂的人际关系形成了一个个具有不同属性的社区，也因此衍生出了基于图的社区发现等算法。在搜索引擎应用场景中，可以用图来表示网页之间的超链接关系。为了衡量特定网页相对于搜索引擎索引中的其他网页的重要程度，Google提出了PageRank算法。在交通领域的应用场景中，可以用图来表示车辆或行人的出行轨迹。收集的位置信息构成的图，可以用来在动态的交通网络中查找最短路径，或进行行程规划等。

社交网络、搜索引擎以及交通领域等产生的数据可以用图来表示其中的关联关系。图中的点和边存在着一定的关联性，为了挖掘其中有价值的信息，需要对图数据进行一系列的操作。这些常见场景中的大规模图数据处理已经成为了大数据系统需要支持的主流应用。目前Spark中的GraphX以及Flink中的GElly等都提供了对图算法的支持。

* 机器学习

机器学习[27]是一门人工智能的科学，通常可以分为监督学习、无监督学习和增强学习三类。（1）监督学习是从给定的训练集中学习出一个函数，当有新的数据到来时，可以根据这个函数预测结果。监督学习的训练集包括特征和目标，并且其中的目标是由人标注的。常见的监督学习有回归分析、统计分类等算法。（2）无监督学习中，训练集没有人为标注。常见的无监督学习包括[聚类](https://zh.wikipedia.org/wiki/%E8%81%9A%E7%B1%BB)、降维等算法。（3）增强学习通过观察来学习动作，是一种试错学习。由于没有直接的指导信息，因此，学习对象需要不断的与环境进行交互，通过试错的方式做出判断。

目前，Spark中的MLlib以及Flink中的ML都对机器学习中应用比较广泛的模型（如聚类、回归等）提供了支持。

* 流式应用

随着信息技术网络化、智能化的发展，以及数据规模大、来源丰富、类型复杂、变化迅速等诸多特征，使得高时效、可扩展的数据处理成为保障大数据系统服务质量的必要条件。当前，以智慧城市、智能装备、智能制造、极端交易等为代表的具有大数据、实时流式处理等特征的数据处理场景大量涌现。

流式应用可以从三个维度进行表征[20]：延迟，吞吐量和状态大小。流式应用对大数据系统的处理速度以及处理延迟都有较高的要求。目前，Spark Streaming以微批的方式处理流式数据，但有一定的处理延迟；Flink提供的DataStream API为流式应用提供了接口支持。

## 大数据系统应用可靠性问题

### 可靠性定义

ISO 9126[][]定义了六个质量特性，Reliability（可靠性）就是其中之一，并且包含了以下子特性：Maturity，Fault Tolerance，Recoverability。同时定义可靠性为[]“软件产品在规定的时间内、在规定的条件下，保持其性能水平的能力”（”The capability of the software product to maintain its level of performance under stated conditions for a stated period of time.”）。IEEE 610.12-1990[28][]将可靠性定义为“系统或组件在规定的时间内、在规定的条件下，执行其所需功能的能力”（“The ability of a system or component to perform its required functions under stated conditions for a specified period of time.”）。Gillies等人[]认为由于每个系统以及系统的不同部分的目的是不同的，因此评估可靠性的方式也可能会有变化。

根据《Software Engineering A practioner’s Approach》[]一书中提出的对可靠性的看法，本文总结可靠性为以下三个方面：系统故障频率（Frequency of system failure）、无错系统功能（Error-free system functions）和存在故障或软件错误（Presence of faults/software bugs）。对于大多数的项目和软件开发者来说，可靠性等于正确性[28]，即它们正在测试并发现和修复 “bug”的数量。然而，在大数据系统测试环境下，本文主要关注的是系统是否存在故障或软件错误，而不是系统故障频率。系统是否可靠可以看做系统在特定配置下能否正常运行，即是否会发生执行出错或得不到正确结果的情况。这些错误不单纯是代码的逻辑错误引起的“bug”，可能是应用本身（如，不恰当的配置、数据异常以及用户代码缺陷等）造成的，也可能是系统框架（如，Spark、Flink等）执行机制引起的。

在上述基础上，本文将可靠性定义为“在给定应用（包括数据、代码、参数配置等）的情况下，大数据系统系统在指定的数据和参数配置下，系统是否存在运行时错误、数据完整性或计算结果错误等故障或软件错误”。

本文将在下一节对大数据系统面临的可靠性问题展开进一步的分析。

### 可靠性问题分析

当前Hadoop、Spark、Flink等分布式处理框架被广泛应用于大数据的处理分析场景中，如Web索引建立、日志挖掘、机器学习、大规模图分析等。在这些处理场景中，大数据系统应用通常可以表示为（input data，configurations，user code）。其中，input data表示输入数据，通常作为数据块存储在分布式文件系统上；configurations表示配置参数，包括系统参数（例如，输入块大小，分区号）和应用参数（例如，K-means的集群k）等；user code表示用户代码，即用户定义的函数（如*map()*，*reduce()*和*join()*等），它们用于处理输入数据或中间结果。

然而，这些应用在处理大数据时，经常会出现内存溢出、I/O异常、任务超时等运行时错误；同时，在流处理过程中会遇到数据完整性和计算结果错误等情况。这些运行时错误以及计算结果错误等会直接造成应用执行失败，甚至更加严重的后果。大数据系统应用的可靠性已经成为学术界和工业界尤为关注的问题。

已有的关于大数据系统的可靠性研究主要有以下几点：

（1）MapReduce应用内存使用问题方面：由Xu[8]等人通过研究123个真实Hadoop和Spark应用的内存溢出错误，发现了内存溢出错误的三大原因：框架暂存的数据量过大、数据流异常以及内存使用密集的用户代码。

（2）大数据在线查询分析错误方面：Li[9]等人研究了250个SCOPE job（运行在微软的Dryad[]框架之上）的故障错误，发现错误主要原因是未定义的列、错误的数据模式以及不正确的行格式等。其中84.5%的错误是由于数据处理中的缺陷引起的。他们也发现3个内存溢出错误，错误原因是在内存累积了大量的数据（比如一个大表的所有行被存放到内存中）。

（3）大数据系统运行错误方面：Kavulya等人[10]分析了4100个执行失败的Hadoop jobs，这些jobs运行在Yahoo!管理的M45[]集群。他们发现36%的故障是数组访问越界错误，还有23%的故障是I/O异常。Zhou等人[29]研究了大数据平台在微软中存在的问题，他们发现36％的问题是由系统缺陷引起的，其中2个问题（1％）是内存问题。

（4）系统在云环境中的开发和部署方面：Gunawi等人[30]研究了云上部署的系统（如Hadoop MapReduce、HDFS和HBase等）的3655个issues。他们发现87％的问题是软件故障，而13％的问题是硬件故障。他们还报告了HBase中的1个内存溢出错误（用户在大型数据集上提交查询）和Hadoop文件系统中的1个内存溢出错误（用户并行创建了数千个小文件）。

上述的研究发现表明，这些错误的原因多种多样，既包括系统缺陷（如设计缺陷和实现bugs），也包括数据异常（数据维度过高、数据倾斜等）和应用缺陷（如参数配置不当、代码缺陷等）。目前，针对这些错误的解决方案[14][15]通常是在问题出现之后，再针对某一类应用及问题进行分析诊断。通过测试来发现错误是一种常用的方法，由于运行时错误的多样性和复杂性，当前还没有一个通用的错误检测方法能够提前发现系统、应用和数据的潜在问题。

## 测试基准框架研究现状

当前，常见的针对大数据系统的测试基准有BigBench[31]、BigDataBench[32]、HiBench[16]和SparkBench[20]等；针对特定的大数据应用的测试基准包括Spark SQL benchmark[17]（SQL查询）、Graphalytics[19]（大规模图分析）、StreamBench[33]（流式应用）等。

* 大数据系统测试基准

BigBench是一种端到端的大数据测试基准，其基本业务模式是产品零售商。该测试基准涵盖了一个数据模型和一个合成数据生成器，针对数据模型的一组查询，生成具有更大数据集、更高速率、更多数据类型（如，来自关系表的结构化数据、半结构化数据以及来自网络点击和社交媒体中的非结构化数据等）的合成数据。BigBench的应用场景比较单一，对大数据系统中的应用类型覆盖不全面。

BigDataBench是针对Web搜索引擎提供大数据测试基准，其相对于BigBench涵盖了更广泛的应用场景，同时提供了多样化和有代表性的真实数据集。该测试基准提供了6个真实数据集以及19个大数据工作负载，并针对不同的数据类型和数据来源提供不同的方法来合成数据。BigDataBench提供的不同类型的工作负载（如社交网络、搜索引擎、关系查询等），分别对应于不同的大数据系统（如Spark、Hadoop、MySQL等）。

HiBench是针对Hadoop系统提出的测试基准。该基准提供了微基准、Web搜索、机器学习等8个工作负载，并从作业运行时间、吞吐量（每分钟完成的任务数）、HDFS带宽以及系统资源（如CPU、内存及I/O等）等方面来评估Hadoop性能。同时，HiBench还为Hadoop的测试提供了非结构化类型的真实数据集。

SparkBench是针对Spark的性能测试工具。BigBench，BigDataBench和HiBench等其他测试基准仅涵盖了少量的Spark支持的应用类型。SparkBench测试基准则覆盖了Spark支持的四种主流的应用类型，包括SQL查询、大规模图计算、机器学习以及流式应用。并且测试中使用到的数据大部分由自带的数据生成器生成。通过对算法或应用的横向对比以及纵向对比测试，从系统资源（如，CPU、Memory等）、任务执行时间、数据处理速度等方面考察Spark的性能。

* 大数据应用测试基准

Pavlo[34]设计了一个SQL基准测试来比较MapReduce和关系数据库之间的性能。Berkeley AMPLab开发了一个SQL基准[17]来比较Spark、Apache Hive[]以及Apache Impala[]等的性能。

Graphalytics是专门用于图处理平台的测试基准。该测试基准提供了有代表性的图计算应用，同时使用LDBC社交网络基准（SNB）数据生成器（Datagen）生成的Graphalytics合成数据集。目前，已经测试了比较流行的Giraph，GraphX和Neo4j等图处理平台。

StreamBench是一个用于流处理系统性能比较的基准框架。该测试基准定义了一个公共API组件和一组核心工作负载，并为三种广泛使用的开源流处理系统（Storm、Flink和Spark Streaming）提供基准测试。StreamBench支持定义新的工作负载，同时支持对新的流处理系统进行基准测试。

* 测试基准框架存在的问题

表2-2总结了上述介绍的大数据系统及应用的测试基准的特性。通过对这些测试基准框架进行分析，发现现有的测试基准框架主要存在如下问题：

（1）仅支持性能测试。

现有的测试基准仅针对大数据系统应用的性能提供了基准测试，没有考虑大数据系统应用存在的可靠性问题。

（2）仅提供常规测试。

现有的测试基准通常使用常规数据以及固定的配置参数来测试大数据系统，缺乏测试的多样性。

表格 2-2 测试基准比较

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **测试基准** | **数据集** | **工作负载** | **测试对象** | **测试类型** |
| **BigBench** | 合成数据集 | 关系查询 | DBMS、Hadoop | 性能测试 |
| **BigDataBench** | 真实数据集、合成数据集 | 社交网络、搜索引擎、关系查询 | NoSQL系统、实时分析、在线分析系统 | 性能测试 |
| **HiBench** | 真实数据集 | 微基准、Web搜索、机器学习 | Hadoop、Hive | 性能测试 |
| **SparkBench** | 真实数据集、合成数据集 | SQL查询、图计算、机器学习以及流式应用 | Spark | 性能测试 |
| **Graphalytics** | 真实数据集、合成数据集 | 大规模图计算 | GraphX、Giraph、Neo4j | 性能测试 |
| **StreamBench** | 合成数据集 | 基本操作、join、迭代 | Storm、Flink、Spark Streaming | 性能测试 |

## 可靠性测试基准框架需求

针对大数据系统及应用面临的可靠性问题，以及现有测试基准存在的不足，面向大数据系统及应用的可靠性测试基准框架有很大的研究价值和发展空间。Huppler[35]提出，一个成功的基准需要具备的属性有相关性、可重复性、公平性、可验证性、经济性等。随着大数据系统的迅速发展，Agrawal[20]等人补充测试基准的新属性有如下几点：

（1）简单易用。测试基准需要易于理解和部署，并能通过适度的配置便可自动化的执行和分析。

（2）应用多样性。测试基准需要满足大数据系统中应用类型的多样性，通过选择不同操作类型的有代表性的应用来体现大数据系统操作的多样性。

（3）可移植性。测试基准需要适配更广泛的大数据系统，能够为更多的大数据系统提供测试。

（4）可扩展性。测试基准需要满足大型分布式系统的测试需求，能够生成数据量大且变化多样的数据集。

可靠性测试基准框架同样应该满足上述需求，同时还能够解决上一节中提到的现有测试基准的不足之处。因此，可靠性测试基准框架还应满足以下几点需求：

（1）测试数据多样性。为了发现大数据系统可能存在的问题，测试基准需要提供更加丰富多样、能够满足应用出错特征的数据集。

（2）测试多样性。为了更加方便快速的发现潜在的可靠性问题，测试基准需要提供满足不同配置的、更加多样的测试。

（3）负载突发性。对于流式应用，可靠性测试基准需要提供流速可变的流式负载，并通过突发性的负载发现大数据系统可能存在的系统或应用错误。

# 可靠性测试基准设计

为了能够提前发现大数据系统应用存在的可靠性问题，本章设计了大数据系统应用的可靠性测试基准。首先，通过实证分析，给出了可靠性测试基准的设计需求；然后，介绍了基准应用中的应用类型及工作负载；其次，介绍了基准提供的测试数据；接着，介绍了基准执行过程；最后，介绍了可靠性测试基准的度量指标。

## 基准设计需求

本节通过对issues以及相关论文进行实证分析，在表3-1中总结了现有大数据系统中存在的可靠性问题以及出现这些问题的原因。

图表 3-1

|  |  |  |  |
| --- | --- | --- | --- |
| 错误 | 内容 | 原因 | 来源 |
| 内存溢出 | Xu[8]等人研究了123个真实Hadoop和Spark应用的内存溢出错误。 | 发现主要原因是：框架暂存的数据量过大，数据流异常以及内存使用密集的用户代码。 | 论文 |
| I/O异常 | Kavulya[10]等人通过分析4100个执行失败的Hadoop jobs，发现23%的故障是IO异常。 | Zhou等人[29]研究了大数据平台在微软中存在的问题，他们发现36％的问题是由系统缺陷引起的。 | 论文 |
| 运行超时 | Li[9]等人研究了250个SCOPE job（运行在微软的Dryad框架之上）的故障错误。 | 发现错误主要原因是未定义的列，错误的数据模式，不正确的行格式等。 | 论文 |
| 计算结果错误 | Spark SQL中，一个表同时参与多次join操作。 | 当一个表同时参与多次join操作但不重新命名时，Spark会将列名弄混导致结果错误。 | Spark-11576 |
| GC | LogisticRegression中aggregator序列化不必要的数据。 | 由于大量的序列化不必要的数据，将导致性能问题（如频繁GC）。 | Spark-16008 |

通过上述实证分析，发现大数据系统出现可靠性问题（运行时错误、计算结果错误等）的原因多种多样。本文总结产生上述问题的原因为以下几点：

1. 系统缺陷：如设计缺陷和实现bugs。
2. 数据异常：如数据格式错误、数据倾斜等极端的数据。
3. 应用缺陷：如配置参数不当、代码缺陷等。

由上述产生可靠性问题的原因发现，大数据系统中的错误一般是在极端配置、极端数据下产生的。因此，可靠性测试基准为了能够检测出大数据系统中存在的这些可靠性问题，需要满足以下几点需求：

1. 典型应用选取：选取的应用需要能够代表大数据系统应用类型的计算特征。
2. 异常数据生成：测试基准需要提供更加丰富多样、能够满足应用计算特征的极端异常数据。
3. 异常参数配置：测试基准需要提供满足不同配置的、更加多样的测试。

## 基准应用

可靠性测试基准的构建首先需要选取基准应用。本节将介绍根据计算特性选取的大数据系统应用中使用广泛且有代表意义的典型应用；同时，介绍了不同应用类型的工作负载。

### 应用类型

现有的大数据应用场景主要包括社交网络、搜索引擎、数据采集及数据查询等，并根据这些应用场景衍生出了一系列用于处理特定领域的典型应用。本文针对这些应用场景，并依据现有的大数据系统测试基准中提供的应用类型[16,36-39]，选取了SQL查询（SQL）、大规模图分析（Graph）、机器学习（Machine Learning）中的若干应用作为可靠性测试基准的基准应用。表3-2列出了可靠性测试基准选取的基准应用类型及其计算特性。

### 工作负载

本文针对上述基准应用类型，分别在SQL、Graph、Machine Learning中提供了下述具体的工作负载。

* SQL

SQL是一种使用广泛的查询语言，并有广泛的大数据生态系统支持SQL查询。现有的数据库评测基准，如Big Data Benchmark，TPC-H和TPC-DS等，都覆盖了较为丰富的查询语句。这些应用于不同的大数据系统，甚至不同应用领域的SQL查询，通常可以由基本的查询子句构造出来。

表格 3-2 可靠性测试基准的基准应用

|  |  |  |
| --- | --- | --- |
| **类别** | **应用** | **计算特性** |
| **SQL** | Scan | 单表过滤操作 |
| Aggregate | 单表聚合操作 |
| Join | 多表关联操作 |
| Mix | 多表混合操作 |
| **Graph** | PageRank | 迭代计算 |
| TriangleCount |
| ConnectedComponents |
| SingleSourceShortestPaths |
| **Machine Learning** | LogisticsRegression | 分类算法、迭代计算 |
| K-means | 聚类算法、迭代计算 |
| ALS | 交替最小二乘法 |
| RandomForest | 分类/回归、宽度优先树 |
| SVM | 分布式双梯度下降 |
| **Streaming** | WindowJoin | 流式聚合 |

表格 3-3 表结构

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | 属性名 | 数据类型 | 含义 |
| Rankings | pageURL | VARCHAR | 网页URL |
| pageRank | INT | 网页排名 |
| avgDuration | INT | 平均停留时间 |
| UserVisits | sourceIP | VARCHAR | 源IP地址 |
| destURL | VARCHAR | 目标URL地址 |
| visitDate | DATE | 访问日期 |
| adRevenue | FLOAT | 广告收入 |
| userAgent | VARCHAR | 用户代理 |
| countryCode | VARCHAR | 城市编码 |
| languageCode | VARCHAR | 语言编码 |
| searchWord | VARCHAR | 搜索关键字 |
| duration | INT | 停留时间 |

为此，本文根据Pavlo[34]等人对测试基准的研究，参照其测试方案设计了两张表，分别为Rankings和UserVisits。其中，Rankings表用于记录网页排名信息，UserVisits表用于记录用户访问网页信息。两张表的属性以及具体含义如表3-3所示。以下是本文选取的SQL查询中的工作负载。

Scan

针对单个二维表，提供基本过滤查询，如SELECT *\** FROM *uservisits* WHERE *adRevenue > X;*

Aggregate

针对单个二维表，提供聚合操作，如SELECT *destinationURL*,sum(*adRevenue*) AS *total* FROM *uservisits* GROUP BY *destinationURL* ORDER BY *total* DESC;

Join

针对多个二维表，提供多表关联操作，如**SELECT** *sourceipaddr*,*url*,*adrevenue* **FROM** *rankings* **INNER** **JOIN** *uservisits* **ON** *url = destinationURL;*

Mix

针对多个二维表，提供过滤、聚合及关联的混合操作，如SELECT *sourceIPAddr*, *totalRevenue*, *avgPageRank* FROM (SELECT *sourceIPAddr*, AVG(*pageRank*) as *avgPageRank*, SUM(*adRevenue*) as *totalRevenue* FROM Rankings AS *R*, Uservisits AS *UV* WHERE *R.url* = *UV.destinationURL* AND *UV.avgTimeOnSite* BETWEEN 30 AND 70 GROUP BY *UV.sourceIPAddr*) ORDER BY totalRevenue DESC LIMIT 10。

针对上述SQL中的基准应用，可靠性测试基准通过提供规模不同、倾斜程度不同的表文件来测试SQL应用的可靠性。

* Graph

社交网络中人与人之间相互通信的关系、交通网络中车辆的行驶轨迹等都可以用图来表示。现有的大数据系统中，Spark的GraphX以及Flink的GElly等都提供了图应用。本文依据现有的大数据系统应用库中提供的图算法，以及现有的大数据系统测试基准中提供的图应用，选取了如下几种典型应用作为可靠性测试基准的工作负载。

1. PageRank

PageRank是Google提出的在搜索结果中对比网站排名的算法，通常应用于搜索引擎中对搜索结果排序的场景下。该算法在图计算中，通常使用迭代模型计算。例如，使用顶点为中心的迭代模型进行PageRank计算过程中，图中的每个顶点在每次迭代过程中都会从相邻顶点处接收消息，并在经过计算后，活跃的顶点会将更新后的消息传递给邻居顶点。持续的进行迭代，直到达到最大迭代次数或结果收敛。

1. TriangleCount

TriangleCount用于统计（有向/无向）图中不同三角形的数目。一般运用在社交网络分析中。社交网络中的三角形越多，说明关系网越强。该算法在图计算中同样使用迭代计算模型。

1. ConnectedComponents

ConnectedComponents用于计算输入的图的连通分量。在一个图中，如果将顶点抽象为变量，顶点之间的连通性抽象为变量间的引用关系，那么构造并查集就可以解决变量的循环引用问题，避免引用环的存在，从而方便引用技术和垃圾回收。同样的，如果将顶点抽象为计算机，顶点之间的连通性抽象为网络，那么动态连接性问题可以减少不必要的布线。

1. SingleSourceShortestPaths

SingleSourceShortestPaths用于计算图中单源最短路径。该工作负载可以评估大数据系统中的图处理框架是否可以有效地处理本地或子图计算。

对Graph中的基准应用进行测试时，首先需要生成图数据到存储系统，然后Graph中的基准应用会以该数据为输入进行测试。Graph应用的测试通过提供规模不同或顶点分布不同的图数据文件来测试Graph应用的可靠性。

* Machine Learning

Machine Learning在大数据处理分析中应用较为广泛，为结果预测、信息推荐等提供了模型和算法支持。目前使用较为广泛的算法有回归、分类、聚类等。现有的大数据系统中，Spark的MLlib以及Flink的ML都提供了相应的算法实现。本文依据现有的大数据系统中的机器学习算法以及流行的大数据性能测试基准中提供的机器学习应用，构建了如下工作负载。

1. LogisticsRegression

LogisticsRegression作为机器学习的分类器，可用于预测连续或分类数据。该算法使用随机梯度下降或L-BFGS来训练分类模型。输入的数据集在每次迭代中计算、更新和广播参数向量。

1. K-means

K-means是一种常见的聚类分析算法，使用无监督学习方法，通过迭代不断的选取离中心点最近均值进行聚类分析。

1. ALS

ALS算法将稀疏评分矩阵分解为用户特征向量矩阵和产品特征向量矩阵的乘积，并使用交替最小二乘法算法来计算矩阵分解。

1. RandomForest

RandomForest是基于决策树的集成模型，通过对数据随机采样来单独训练每一棵树；通过组合每棵独立树的结果进行预测；通过训练数据来构建一棵回归树，从而对未知数据进行回归预测。

1. SVM

SVM模型使用基于hinge-loss的具有高效通信能力的分布式双梯度上升的方法，通过在高维或无限维空间中构建一组超平面来进行分类训练。与线性和逻辑分类相比，SVM可以将输入隐式映射到高维特征空间，并有效地进行非线性分类。

Machine Learning的可靠性测试通过输入不同的数据集（如，数据规模、数据分布、数据维度等不同），同时针对不同的基准应用提供不同的参数组合，来进行参数组合测试。

* Streaming

随着智慧城市、智能装备、智能制造、极端交易等大数据处理场景的涌现，越来越多的数据需要实时地、流式地处理，以满足数据的时效性。本文主要选择了流式应用中使用比较普遍的WindowJoin作为流式工作负载。WindowJoin提供了两个使用窗口模式的数据流的join操作。与批处理join操作不同的是，流式处理的join操作会受到流速以及流量的影响。针对Streaming的基准应用，通过提供不同流速及流量的并发流式负载，来测试流式应用的可靠性。

## 测试数据

本节将针对上一节提出的基准应用提供测试数据，并从SQL、Graph以及Machine Learning三种应用类型给出测试数据的设计。另外，本文将在？节详细的介绍数据生成方法。

* SQL数据

SQL数据根据3.3.2节中给出的Rankings表和UserVisits表定义的具体模式来生成。Rankings表中的数据格式为（pageURL,pageRank,avgDuration）。UserVisits表中的数据格式为（sourceIP,destURL,visitDate,adRevenue,userAgent,countryCode,

languageCode,searchWord,duration）。其中，visitDate，adRevenue和sourceIP字段从特定范围内生成随机值，而其他字段（如countryCode、languageCode等）将从真实数据集进行随机选择。

SQL的数据生成通过配置相应的参数生成常规数据或倾斜的异常数据。数据通过集群分布式的生成，且每条数据之间以换行作为分隔，每个数据文件以文本文件的形式存储在HDFS的节点上。

* Graph数据

Graph数据包括Vertex（顶点）数据集以及Edge（边）数据集。Vertex定义为二元组(vertexID,value)，其中vertexID用来标识顶点的编号，value是顶点的值。Edge定义为三元组(srcID,destID,value)，其中srcID为源顶点ID，destID为目的顶点ID，value为源顶点到目的顶点的边的权重。Vertex中的vertexID采用范围内顺序生成的方式，value为满足某个范围的随机数据。Edge中的srcID和destID需要在Vertex的vertexID范围内生成，其权重value为特定范围内生成的随机值。

Graph的数据生成可以通过配置顶点之间的相遇概率，生成满足不同幂率分布的数据。数据文件以文本文件的形式存储在HDFS节点上。

* Machine Learning数据

Machine Learning中不同应用的数据格式有所不同。

LogisticsRegression、RandomForest以及SVM都是有监督的学习。这些应用使用带标签的数据，数据的格式为。这种数据格式称为LibSVM Format。其中label表示标签，在分类算法中为离散值（例如，二分类时，label取值为0或1），在回归算法中为连续值。index表示各个特征的序列号，value表示各个特征值。在监督学习中，各个样例不一定包含全部特征，因此数据集通常为稀疏数据。

K-means是无监督学习，使用不带标签的数据。数据的格式为。value表示各个特征值。无监督学习中，每个样例都包含全部特征，因此数据集通常为稠密数据。

ALS数据使用Netflix数据集（电影评分数据集）格式。其中userId为用户编号，itemID为电影编号，rating为电影评分，timestamp为时间戳。userId和itemId为某一范围内的随机值，rating为1~5范围内的随机数，timestamp为日期类型的数据。

Machine Learning的数据生成同样需要配置数据特征参数（如实例数、维度、分布方式等），并根据配置信息生成满足不同维度、不同分布、不同规模的数据。这些数据同样存储在HDFS的节点上。

## 基准执行

在具备了基准应用和测试数据后，本节将介绍可靠性测试基准的执行流程，如图？所示。

（1）在准备好待测系统以及存储系统的环境后，用户首先在可靠性测试基准中配置系统信息，包括待测系统和存储系统的访问路径等信息。

（2）测试环境以及访问路径都准备好后，用户通过配置数据生成需要的参数信息，可以生成自定义的数据集。例如，在SQL数据生成中，可以配置数据倾斜度来生成倾斜的异常数据。

（3）数据生成结束后，用户选择工作负载，并配置所需的系统参数和应用参数信息。同时使用已经生成的数据集，通过触发脚本执行集群环境下的参数组合测试。

（4）测试完成后，用户可以查看测试报告。



图表 3-1 基准执行阶段

## 测试度量

在性能测试基准中，经常以吞吐量、响应延迟作为度量指标。在软件运行的可靠性测试中，已有的方法通常使用故障平均时间来衡量系统的可靠性[28]。由于大数据系统运行在多节点的集群环境下，系统的可靠性无法使用常规的故障平均时间来衡量。本文将以下异常现象称为系统不可靠的评测标准：

1. 性能出现异常。如，在用户给定的时间内，出现无响应、假死等现象。
2. 资源使用异常。如，出现IO异常、内存溢出、磁盘异常等现象。
3. 计算结果异常。如，计算结果错误、计算数据丢失或重复计算等现象。

由上述系统不可靠的评测标准，本文给出的大数据系统可靠性测试基准的度量指标为：在用户给定的系统配置下，如果出现上述异常现象，则称该系统在给定的系统配置下是不可靠的。