清 华 大 学

综 合 论 文 训 练

题目：基于分布式流式计算的
电力大数据分析

系　　别：自动化系

专　　业：自动化

姓　　名：赵兵兵

指导教师：曹军威 副研究员

2014 年 6 月 20 日

关于学位论文使用授权的说明

本人完全了解清华大学有关保留、使用学位论文的规定，即：学校有权保留学位论文的复印件，允许该论文被查阅和借阅；学校可以公布该论文的全部或部分内容，可以采用影印、缩印或其他复制手段保存该论文。

(涉密的学位论文在解密后应遵守此规定)

签　名：_____导师签名：_____日　期：_____

中文摘要

伴随着智能电表的普及和物联网等新型 IT 技术在电力行业中广泛应用，电力大数据在数量和质量上急剧增长。同时，大规模分布式存储和开源分布式计算框架为电力数据的记录和分析提供了基础。由于电网的很多应用对时延有很高的要求，传统的基于 Hadoop 的批处理很难满足需求，因此我们提出利用分布式流式计算来搭建电力大数据的框架。本文首先介绍了电力大数据产生的背景，以及数据在电力行业的未来扮演的重要角色。接下来是电力大数据基本架构，包括数据采集、存储、分析与挖掘、应用。第 3 章介绍了当前主流的流式计算平台 Storm，并且比较了它相对于 Hadoop 和其他流式平台的优势。最后，我们利用 Storm 进行电网电能质量中电压暂降的分析。包括多源异构的数据流聚合；电压暂降可视化；利用支持向量机对不同原因导致的电压暂降事件进行分类。

关键词：智能电网；电能质量；大数据；流式计算；实时性

ABSTRACT

With smart meter coming to every home and the Internet of Things being applied in power industry, the data of power grid is increasing dramatically and acquired value which cannot be obtained in small scale. At the same time, large-scale distributed storage and open source distributed computing provide a platform for record and analysis of big data in smart grid. Because many applications in smart grid have high demand for latency, and traditional batch processing cannot meet this, stream computing is important. This article firstly introduce how power big data is generated and the role of data in power industry. Next, a review of basic framework and data mining algorithm is presented. Then we introduced 2 stream computing platforms, Storm and S4. At the last, we use Storm to do experiments and analysis of power quality.

Keywords： smart grid; big data; framework; storm; stream computing; real-time

目　　录

第1章 引言

1.1 研究背景

“大数据”（big data）是时下 IT 行业最火热的名词。与此同时，数据仓库、数据安全、数据分析、数据挖掘等大数据商业价值也逐渐成为人们追捧的热点。

美国互联网数据中心指出，互联网上的数据每年将增长 50%，每两年便将翻一番，而目前世界上 90%以上的数据是最近几年才产生的[1]。2010 年，全球进入所谓 ZB 时代（1TB=1024GB，1ZB=10 亿 TB）；2020 年，全球数据规模将达到 40ZB。如果把 40ZB 的数据全部存入现在的蓝光光盘，这些光盘的重量相当于 424 艘航母。而对于地球上的几十亿居民而言，这意味着人均拥有 5247G 数据，即人均存满 10 多个 500G 硬盘。

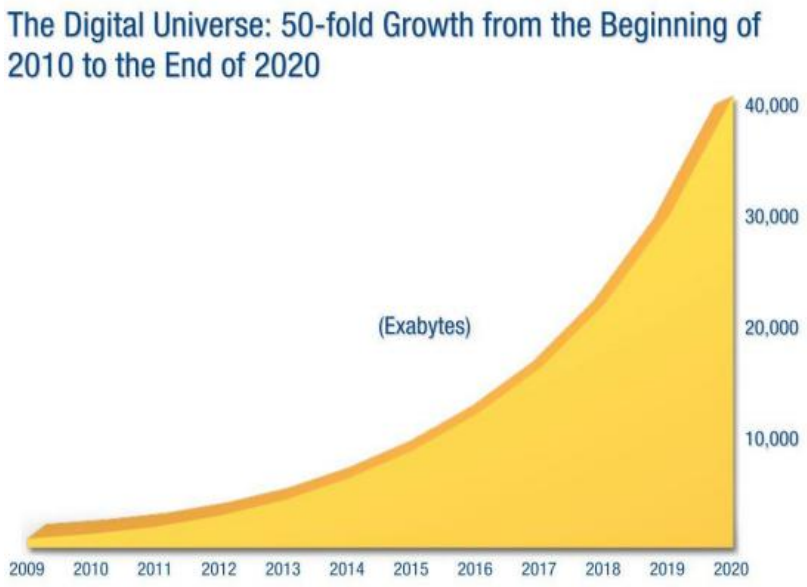


图 1.1 《IDC 数字世界》对全球数据规模的预测

大数据高速发展并且得到重视的原因有三：

一、随着互联网、云、物联网、移动互联网、可穿戴设备的普及，使得数据产生很便宜，很方便。全世界的工业设备、汽车、电表上有着无数的数码传感器，随时测量和传递着有关位置、运动、震动、温度、湿度乃至空气中化学物

质的变化，也产生了海量的数据信息。数据量的爆炸性的增长使得可以做很多小数据基础上无法完成的事情。

二、计算机性价比的提高，磁盘价格的下降，云计算的兴起，Hadoop 的诞生，使得大规模分布式环境搭建的成本异常低廉，大幅降低了大数据存储和处理的门槛。

三、当数据变为知识后，可以帮助企业细分市场、精准营销、优化管理等。随着竞争日益激烈和市场接近饱和，如何能更加了解客户是下一步企业竞争的焦点。著名咨询公司麦肯锡在 2011 年 5 月发布报告：《大数据：创新、竞争和生产力的下一个新领域》[2]：“数据已经渗透到每一个行业和业务职能领域，并逐渐成为重要的生产因素”。

大数据具有以下公认的四大特点：

第一， 数据容量大（Volume）。从 TB 级别，跃升到 PB 级别；

第二， 数据类型繁多（Variety）。相对于以往便于存储的以文本为主的结构化数据，非结构化数据越来越多，包括网络日志、音频、视频、图片、地理位置信息等，这些多类型的数据对数据的处理能力提出了更高要求。

第三， 商业价值高（Value）。价值密度的高低与数据总量的大小成反比。以视频为例，一部 1 小时的视频，在连续不间断的监控中，有用数据可能仅有一二秒。如何通过强大的机器算法更迅速地完成数据的价值“提纯”成为目前大数据背景下亟待解决的难题。

第四， 处理速度快（Velocity）。1 秒定律。这是大数据区分于传统数据挖掘的最显著特征。根据 IDC 的“数字宇宙”的报告，预计到 2020 年，全球数据使用量将达到 35.2ZB。在如此海量的数据面前，处理数据的效率就是企业的生命。

大数据的特点决定了其算法和计算的特殊性。

首先，由于数据容量大，因而数据多采用分布式存储方式，同时由于处理速度的要求，决定了对于大数据应用支持的算法必须包含分布式计算框架。

其次，数据类型繁多，以关系型数据库为代表的结构化数据已经不是全部，甚至不是主要部分，而以自然语言文本、音频、视网、图片、科学观测数据、网络日志、传感器数据及各种特殊文件格式存储的半结构化、非结构化数据成为被处理的数据对象的主体，这就决定了大数据计算需要具备“理解”不同格式存储的数据的能力，换句话而言，先对海量的非结构化数据进行“结构化”

的转化是完成进一步大数据计算的基础。所谓的“结构化”就是用同样的模式理解所有的同类数据，而不一定是转化成二维表那样狭义的结构。

第三，商业价值高决定了大数据计算是价值导向的，而“值密度的高低与数据总量的大小成反比”的规律反应了“价值”在数据中的分布是不平均。因而，对大数据进行计算必须是目标明确的，要紧紧基于应用的目的和价值导向来进行，不能盲目上胡乱算上一气，然后期望能获得什么有用的结果。

最后，处理速度快的要求决定了大数据计算能力达到和数据增长速度相匹配的程度才可以。这一点又由于处理实时性的要求而分为两个部分，一是积累批量数据定期完成计算的一般性的分布式计算；二是数据即到即算的分布式流计算。

1.2 电力大数据

电力大数据的发展可以追溯到电力工业信息化。20 世纪 60 年代，发电侧和输电侧基本实现了自动监测和控制；80 年代到 90 年代，计算机辅助设计、电网调度自动化、计算机仿真系统、、电力负荷控制等一批技术被引入；进入 90 年代后，电力企业向企业信息化方向发展，电力企业建立自己的管理信息系统，实现企业系统集成和信息化管理。

进入新世纪以来，伴随着电网中的监控节点越来越多、采集频率越来越高，电力数据规模呈现指数式快速增长，电力企业也纷纷建立数据中心进行储存和分析。

当电力数据规模增大以后，我们可以洞悉一些小规模数据所不具有的潜在的价值。用户行为的挖掘可以让我们制定更好的营销策略。同时，我们可以通过大数据分析出输入输出的关联，从而分析出电网运行的一些规律。我们可能不知道这些规律的因果关系，但是我们还可以通过大数据先拿到结果，然后通过这些结果来推测和研究为什么是这样的。

国家电网对电力大数据的定义是：“电力大数据是以业务趋势预测、数据价值挖掘为目标,利用数据集成管理、数据存储、数据计算、分析挖掘等方面核心关键技术，实现面向典型业务场景的模式创新及应用提升。”[3]电力行业作为一个传统行业，也由于经济的转型和增长的放缓而面临发展的瓶颈。传统的电力行业如何在信息化做主流的今天，发现新的需求，优化管理，降低成本，开发环保、可持续发展的新型能源模式。

1.3 分布式计算

分布式计算是利用网络把成千上万台计算机连接起来，组成一台虚拟的超级计算机，完成单台计算机无法完成的超大规模的问题求解。它把一个需要非常巨大的计算能力才能解决的问题分成许多小的部分，然后把这些部分分配给多台计算机进行处理，最后把这些计算结果汇总得到最终的结果[4]。

分布式计算研究主要集中在分布式操作系统研究和分布式计算环境研究两个方面，在过去的 20 多年间出现了大量的分布式计算技术，如中间件技术，网格技术，移动 Agent 技术，P2P 技术，以及最近推出的 Web Service 技术等。每一种技术都得到了一定程度的认同，在特定的范围内得到了广泛的应用[5]。

1.3.1 网格计算

分布式计算技术迈入到网格计算时代是一次质的飞跃。网格计算的概念最早由美国阿贡国家实验室的 Ian Foster 提出。被定义为“动态的、多虚拟实体之间的资源协同共享以及合作解决问题”[6]。网格环境中的虚拟组织的概念在文献[]中得到了延伸，其首创性的提出了描述网格五层沙漏模型。类似于 TCP/IP 协议的基本框架，该模型的目的在于利用其搭建具体应用。除了单纯的融合学术界思想以外，网格技术的发展也吸取了一系列工业标准，以服务为中心的网格体系结构 OGSA 随即提出，该体系结构建立在网格技术与 Web Service 相融合的基础上。此体系将所有的资源都被建模成服务，于是服务计算（Service Computing）也伴随着网格的兴起而逐步演变成服务科学[7]。

网格计算是在高性能计算时代衍生出的技术，其关注的性能指标依然是每秒钟计算的浮点次数，即目标是将具有复杂计算需求的应用的计算时间降低。而随着云计算时代的到来，人们的关注点转移到高通量计算上。换句话说，云计算技术的目标是实现海量应用和请求具有高并发度和良好整体处理能力，并不需要关注单个应用本身。

1.3.2 云计算

狭义云计算是指计算机基础设施的交付和使用模式，指通过网络以按需、易扩展的方式获得所需的资源（硬件、平台、软件）。提供资源的网络被称为“云”。“云”中的资源在使用者看来是可以无限扩展的，并且可以随时获取，按需使用，随时扩展，按使用付费。

广义云计算是指服务的交付和使用模式，指通过网络以按需、易扩展的方式获得所需的服务。这种服务可以是计算机和软件、互联网相关的，也可以是其他的服务。云计算是虚拟化(Virtualization)、效用计算(Utility Computing)、IaaS(基础设施即服务)、PaaS(平台即服务)、SaaS(软件即服务)等概念混合演进并跃升的结果。

云计算平台的主要特征用以下五个关键词可以很好的概括[8]：

（1）租用（leased）：云计算资源使用方式以租用而非购买为其主要特性。构建数据中心的成本是巨大的，而对于业务和任务需求不太稳定和确定的企业而言，构建数据中心本身具有很大的盲目性。比如有的企业运行应用的工作负载随着时间变化具有很大的波动性和峰值特性，而有的企业的工作负载则相对平稳，有的企业的工作负载具有很好的可预测性，而有的企业则非常不可预测。基于此，云计算中的租用方法则能够很好的迎合以上各种工作负载的需求。

（2）弹性：云计算资源可以按需要实现数量的自由伸缩。使得计算系统适应于工作负载变化，从而自动的增加和减少计算资源的一种方式。动态提供的资源能够及时响应工作负载的变化，而也能很好的匹配工作负载对于计算资源的需求。

（3）定制：云计算节点可以根据应用需求提供动态多样的计算资源。

（4）按需：云计算节点不需要任何时候都不间断运行，只需要按照需求启动或者停止计算资源即可。

（5）灵活：各种计算资源可以实现非常自如的混搭（Mashup），各种服务也可是自如的使用。

1.4 数据挖掘

数据挖掘（data mining）是指从海量的数据中通过算法，提取出隐藏关系、趋势和模式的过程，把数据转化成有价值的信息和知识。数据挖掘是一门融合了人工智能、数据库技术、模式识别、机器学习和数据可视化技术的交叉学科，是大数据分析的理论核心[9]。

国际权威的学术组织 the IEEE International Conference on Data Mining 于 2006 年 12 月评选出了数据挖掘领域的十大经典算法（Top 10 algorithms in data mining）[10]。它们分别是 C4.5, k-Means, SVM, Apriori, EM, PageRank,

AdaBoost, kNN, Naive Bayes, and CART。下面我们选取一些在电力大数据中比较常用的进行介绍。

C4.5 算法[11]是机器学习算法中的一种分类决策树算法，它是基于 ID3 算法[]进行改进的。C4.5 算法继承了 ID3 算法的优点，并在以下几方面对 ID3 算法进行了改进：用信息增益率来选择属性，克服了用信息增益选择属性时偏向选择取值多的属性的不足；在树构造过程中进行剪枝；能够完成对连续属性的离散化处理；能够对不完整数据进行处理。

k-means 算法[12]是一个聚类算法，首先从 n 个数据对象任意选择 k 个对象作为初始聚类中心；而对于所剩下其它对象，则根据它们与这些聚类中心的相似度（距离），分别将它们分配给与其最相似的（聚类中心所代表的）聚类；然后再计算每个所获新聚类的聚类中心（该聚类中所有对象的均值）；不断重复这一过程直到标准测度函数开始收敛为止。

支持向量机（Support Vector Machine，SVM）[13]，是一种监督式学习的方法，它广泛的应用于统计分类以及回归分析中。它把 n 维实空间中的数据点，通过一个 n-1 维的超平面分开，并且使分隔超平面使两个平行超平面的距离最大化。

Apriori 算法是一种最有影响的挖掘布尔关联规则频繁项集的算法。其核心是基于两阶段频集思想的递推算法。该关联规则在分类上属于单维、单层、布尔关联规则。在这里，所有支持度大于最小支持度的项集称为频繁项集，简称频集。

最大期望（EM，Expectation–Maximization）算法[14]是在概率（probabilistic）模型中寻找参数最大似然估计的算法，其中概率模型依赖于无法观测的隐藏变量（Latent Variabl）。

Adaboost 算法是一种迭代算法，其核心思想是针对同一个训练集训练不同的分类器(弱分类器)，然后把这些弱分类器集合起来，构成一个更强的最终分类器(强分类器)。其算法本身是通过改变数据分布来实现的，它根据每次训练集之中每个样本的分类是否正确，以及上次的总体分类的准确率，来确定每个样本的权值。将修改过权值的新数据集送给下层分类器进行训练，最后将每次训练得到的分类器最后融合起来，作为最后的决策分类器。

K 最近邻(k-Nearest Neighbor，KNN)分类算法[15]，是一个理论上比较成熟的方法，也是最简单的机器学习算法之一。该方法的思路是：如果一个样本在特

征空间中的 k 个最相似(即特征空间中最邻近)的样本中的大多数属于某一个类别，则该样本也属于这个类别。

1.5 电能质量分析

理想的电网信号应该是完美对称的正弦波，然而，一些因素会使其偏离，由此便产生了电能质量问题。IEEE 协调委员会对电能质量（power quality）的技术定义为：合格的电能质量是指给敏感设备提供的电力和设置的接地系统均是适合该设备正常工作的。IEC 标准对电能质量的定义为：电能质量是指供电装置在正常工作情况下不中断和干扰用户使用电力的物理特性[16]。

IEEE 第 22 标准委员会和其他国际委员会采用 11 种指标来衡量电能质量。

第一类是电压幅值问题。断电（interruption）：在一定时间内，一相或多相完全失去电压；电压暂降或电压下跌（voltage sag）：电压持续时间 0.5 周期到 1 分钟，幅值 0.1~0.9p.u.；电压上升（voltage swell）：电压持续时间 0.5 周期到 1 分钟，幅值 1.1~1.8p.u.；瞬间脉冲（transient impulse）：在两个连续稳态之间的一种在极短时间内发生的电压（电流）变化；电压波动与闪变（voltage fluctuation and flicker）：在包络线内电压的规则变动，或是幅值不超过 0.9~1.1p.u.范围的一系列随机变化；过电压（overvoltage）：电压或电流持续时间大于 1min，幅值为 1.1~1.2p.u.；欠电压（undervoltage）：电压或电流持续时间大于 1min，幅值为 0.8~0.9p.u.。

第二类是电压波形问题，主要是包含谐波和间谐波。第三类是频率偏差（frequency deviation）。

随着科学技术及工业的发展，许多自动化程度很高的工业用户对电能质量的要求越来越高。在生产或信息技术企业中，大部分装置和一起都带有基于单片机的数字控制器或电力电子器件，这些电子器件对各种电磁干扰都非常敏感，电网非常轻微的扰动就可以影响这些电子系统的运转，致使其不能正常工作或者功能下降，产品合格率降低，使企业蒙受巨大的损失。

国内外由于电能质量问题，特别是电压暂降问题引起的经济损失和不良社会影响比比皆是。在美国，每年由于电压暂降事件所造成的经济损失高达数百亿美元。在欧洲，由于电压暂降问题引起的投诉占电能质量问题投诉量的 80％以上。在广东省，由于雷击或者主网故障的影响，各地方电网每年都有大量的电压暂降现象。比如深圳市的三星电子等企业，由于经常遭受电压暂降的危害，

每次损失数百万元，该企业向深圳市政府投诉，导致深圳供电局为此承受了很大压力。肇庆电网下面的陶瓷厂也经常在雷击日出现停机等异常现象，每次都导致企业受到重大损失。由此可见，电压暂降的危害性正日益突出。因此，对电能暂降的研究和治理具有重要的意义。

第2章　电力大数据关键技术

电力大数据主要可以分为数据采集，数据集成与存储，数据分析与挖掘，电力大数据应用四个部分。其中最核心的技术是数据集成分类技术、数据储存关联技术、高性能分布式计算技术和数据挖掘技术。数据集成要对来自不同数据源的数据进行处理和转换，形成一个统一的新数据源，进而对外提供数据支持；数据存储主要是通过连接各个数据中心的不同的存储设备，使他们连接起来作为一个整体共同对外提供服务；高性能计算是指将多台终端连接形成集群来提供计算资源；而数据挖掘技术是通过机器学习、统计分析和模式识别的算法，来从海量数据中提取出有价值的规律、趋势和特征。图 2.1 展示了电力大数据分析的基本架构。

图 2.1 电力大数据系统架构

下面我们就电力大数据的四个层次，分别阐述他们的核心技术和在电网中的特点。

2.1 数据采集

电力企业的数据主要分为两类[17]：一是电网运行和设备监控数据；二是电力企业营销和管理数据。第一类数据主要由智能电表和安装在电网中的传感器产生，并实时回传至数据中心。特点是数据量大，结构化强。第二类数据是半结构化数据或非结构化数据，但是对电力企业的经营和管理有着重要的作用。

2.2 数据存储

大数据通过把大量普通 PC 服务器通过 Internet 互联，作为一个整体对外提供存储服务，实现大规模分布式存储[18]。

大规模分布式存储的主要优点有：

大容量：大数据一般可以达到 PB 甚至 EB 级的信息量

可扩展：分布式存储系统根据需求随意任意扩展成百上千台的集群规模。系统的整体性能随着系统规模的大小线性变化

低成本：分布式存储系统可以实现自动容错、自动负载均衡。这使得我们可以利用普通的 PC 来搭建，大大地降低了成本。

高性能：吞吐率(Throughput)提高、时延(Latency)减小；

易用：提供易用的对外接口和完善的监控，运维工具，方便与其他系统集成。

分布式存储的类别主要有分布式数据库、分布式表格系统、分布式文件系统三种类别。分布式数据库采用二维表格存储和组织数据，用于存储结构化数据，提供 SQL 关系查询语言。分布式表格系统的存储对象是半结构化数据，这些数据的存储关系比较复杂。分布式文件系统主要用于存储大量的图片、流媒体等非结构化的数据对象。同时分布式文件系统也常常是分布式数据库和分布式表格系统的底层存储。

2.3 数据分析与挖掘

好的算法是发掘大数据价值的关键一步。数据挖掘算法可以揭示出隐藏在大数据下的有意义的关联、趋势和模式，把海量数据转化成有价值的信息和知识。这些算法主要来自人工智能、机器学习、统计分析和模式识别。

同时，随着数据规模的快速增长和算法的时间复杂度和空间复杂度越来越高，对处理器的性能提出了更高的要求。这就需要充分利用集群实现高性能计算。

分布式计算是把一个复杂的计算任务分解，分配给集群中不同的计算设备进行计算，然后再把每台计算机得到的结果汇集在一起。这样，我们可以充分利用现有的计算资源，同时，系统的架构也非常灵活，可以根据不同的任务调整结构。多台计算机连接之后，相当于一台超级计算机，可以完成相当规模的问题求解。

MapReduce 是 Hadoop 提供的开源分布式编程模型。使用 mapreduce，用户在不了解底层机制时，也可以开发自己的并行应用程序。对于电力大数据工程师，重点是电网数据挖掘的代码逻辑，因此 MapReduce 是开发电力大数据系统的不二选择。

另外，由于电网的大数据分析的很多应用具有实时性的特点，以 MapReduce 为代表的批处理很难满足低时延的要求。流式计算应运而生。典型的流式系统有 Twitter Storm 和 Yahoo! S4。在第 3 章我们会进一步介绍其在电力大数据的应用。

在大数据情况下，只有比较复杂的模型，才能充分发掘海量数据下隐藏的丰富价值。深度学习正是在这种情况下应运而生的。

深度学习通过模拟人类大脑的复杂神经结构，具有更优异的特征学习能力，更好地刻画数据的本质。对于电力大数据，当我们用常规的数据挖掘算法（比如决策树、SVM）无法获得好的结果时，我们可以尝试深度学习。

2.4 电力大数据应用

2.4.1 电力数据与宏观经济

社会的各种经济活动背后都存在着电力消费。电力消费与 GDP 往往具有很强的相关性[19]；由于第一产业（农牧业）和第三产业（服务业）电力消费比较

少，第二产业（工业）电力消费比较大，电力消费也是衡量经济结构转型的重要指标；随着智能电表的普及，个人的电力消费数据的快速回传可以方便地反映出房屋的空置比例。

电力数据结构化强、统计简单、方便获取。如何应用电力数据来预测社会经济活动的变化，进而服务于政策的制定，也是电力大数据的热点之一。

2.4.2 电网监控和维护

电网的安全、稳定、高效运行是生产生活的重要基础。设备状态、电能质量分析、故障分析、暂态稳定性等都可以通过大数据进行分析[]。

2.4.3 电网营销和管理

电力价格预测、负荷预测等可以帮助电力企业优化管理，提高利润，实现信息化。

第3章　流式计算

3.1　流式计算简介

在浩如瀚海的数据中，不仅有离线、静态、结构化的数据，同样也有持续产生的、非结构化的、实时传输的数据。这些“流动”的数据，往往对应着实时计算的需求。比如，网站会根据你上一秒的行为，来定制给你推送的广告；电网监控节点数据要经过迅速的处理和计算，得到控制量，实时反馈的闭环，保证了电网的稳定。

流计算所处理的应用主要针对于流式数据。作为大数据中具有核心组成部分的流式数据的实时分析和处理，则是最具有市场潜力和商业前景的方向。流式数据几乎占据了大数据领域中最大的组成部分，从社会化网络（比如新浪微博，Facebook，人人网等），传感器网络，股市财经数据到机场或者火车站摄像头监控数据等。传统基于批处理的计算架构和数据仓库已经越来越不能适应现代流式数据的处理要求。

当前，无论是国内还是国际市场，对于新型流式数据进行收集、分析、处理和可视化具有非常急迫的需求。美国高智公司在去年和今年连续发布云计算和大数据领域十份前沿专利项目申请需求，其中有两份就是关于流式数据分析和处理的。IDC 发布了美国近两年获得大量融资的大数据初创公司，其中有 12 家就定位在数据流处理领域。

我们要在数据“流动”的过程中，进行复杂的逻辑处理，包括统计分析、数值计算、数据挖掘、机器学习等，并且低延时地输出结果。这是传统分布式批处理很难做到的。因此流式计算是大数据时代非常有前景和实用价值的方向。

流式计算(Stream Computing)来源于实时计算(Real-time computing)。在大数据时代，数据不再昂贵，但由于大规模带来的计算高成本，想从浩如瀚海数据中获取我们想要的信息变得昂贵。而要及时的、快速的信息则更加昂贵。实时计算最重要的一个特性是能够实时响应计算结果，而且是妙极时延。当今有很多应用的数据源是流式的，输出结果是流式的，因此催生了流式计算。我们要在数据“流动”的过程中，进行复杂的逻辑处理，包括统计分析、数值计算、数据挖

掘、机器学习等，并且低延时地输出结果。这是传统分布式批处理很难做到的。

相对于离线计算重视吞吐量，流式计算更加注重延迟。当前的互联网环境下，每天都在存储海量的非结构化数据和结构化数据，这些数据需要在比较短的时间内被处理，否则就会带来非常不好的用户体验。流式计算的特点是：数据是一个个到来，下一个时段到来什么数据是未知的；数据到来的速度也是无法预知且无法控制的；数据是有实效性的，必须及时处理；数据从采集系统到达接入层的顺序是不能保证的；任务永无止境[20]。

经典的离线批处理架构是 Hadoop 的 MapReduce。在没有流式计算以前，现实世界中的很多系统把流式数据分割成固定的小块，然后利用 MapRedeuce 平台处理。这种方法的缺点是时延和分割块的长度成比例。同时，总开销也与分割动作的开销和初始化的时间成比例。数据块越小，时延越小，总开销越大。最合适的数据块长度取决于应用。随着诸如实时搜索、高频交易、社交网络等新引用的出现正在突破传统数据处理系统能力的极限。现在需要一个可以满足不同数据规模的流式计算系统，可以处理高数据速率和海量数量。比如，针对个性化的搜索广告，每秒要处理来自报完用户的数以千计的查询。每个用户的行为和特征可以帮助增加模型的准确度，这也使得不同人看到的广告是定制化的。

计算的任务无非来自于科研和商业。科研要求很高的灵活性，对于不同的领域快速开发不同的算法。而对于业界环境，最重要的要求是可扩展（通过增加服务器来增加吞吐量的能力）。我们设计的流式系统要充分考虑这些问题。

流计算

本章中，我们重点考虑各种计算平台技术，调度算法，并提供一些开源的云计算框架。流计算的研究在传统分布式计算中就已经有很好的涉及，但是在云计算的框架里，流计算被重新提起关注，其主要的方向有如下几个[]：

（1）流式数据计算处理技术

呈现出流式数据的请求对于资源的占用和消耗是不同的。有的请求类需要更多的 CPU 资源，有的需要内存资源，有的需要网络带宽资源，有的需要底层 IO 资源，有的需要多种计算资源。鉴于此，在大规模并发的云平台下，如何合理的将流式数据的请求进行分类，从而将不同类别的流式请求转移到与之相适应的计算集群中是本项目需要研究的问题。

本问题的意义在于节省服务提供商的云平台服务资源的使用成本，帮他们在公有云计算平台的多种可供选择的计算资源中选择性价比最高的资源进行配比，使得用户请求能够通过最合理配置的计算资源来响应。

（2）流式数据编程模型技术

编程模型的设计是引领技术走向商业化的最重要的步骤，目前广泛应用的编程模型（MPI，OpenMP，MapReduce，Pregel，Graphlab 等）的推出都改变了人们对于数据处理的方式和方法，产生的巨大的商业和学术价值。MPI 和 OpenMP 的设计基于节点间通信密集型计算需求，MapReduce 的设计基于大量可分割数据并行化处理的需求，Pregel 基于图形数据的处理需求，Graphlab 的设计基于机器学习应用的处理需求等。目前为止，在云上构建流式数据，同时能够支持将计算资源进行弹性增加或者减少的编程模型并不多见。

基于此，在本方向的研究致力于为云计算平台处理流式数据的过程提供一种通用的编程模型，使得编程人员不必关心框架的实现原理，只用根据业务需要在编程模型的框架中设计处理逻辑，底层流式数据的自动处理，持久化存储，任务/数据的并行化，云计算节点的自适应增加和减少等都可以。

（3）流式数据的挖掘技术

国内国外对于流式数据挖掘技术的研究主要体现在如下几个方面：

a）流式数据的聚类：传统数据挖掘中的聚类分析处理的都是静态数据，而对于动态和流式数据的聚类方法，则需要考虑到按照时间具有到达特性的应用和如何处理更新在既有聚类之上的多次聚类问题；

b）流式数据分类：传统分类方法，例如决策树、分类回归树、支持向量机等建立在静态数据集上居多，而近来对于流式数据的动态分类方法则以更新数据为主要特性，结合历史信息和新的信息获取新的分类规则；

c）流式数据的压缩：即在给定误差范围以内，如何把历史数据压缩到一个小的数据集中，根据数据集的小规模数据能够恢复出原始大规模流式数据，本方向的研究方法有线性和多项式拟合、独立成分分析等；

d）规则发现：传统数据挖掘技术的规则发现多体现在多维度之间的关联规则，而流式数据的规则发现需要选择时间维度作为一个方面，即如何选取有效和稳定的规则，对于流式数据中的特征进行刻画，从而为后续到达的流式数据提供好的分析依据则是本方向的重要研究问题。

3.2 开源流式平台

MapReduce，Hadoop 等开源技术和平台是大数据时代最伟大的发明，它让我们轻松地进行大数据的分析和存储。但是 MapReduce 是批处理问题的经典模型，因此它需要一次性送入所有数据，并且一起输出结果，适合于静态数据的大规模分析。企业对大规模实时数据处理要求越来越多。Hadoop 在应对实时性上的低性能使得流式计算平台应运而生。

但在一些应用场景中，数据是从数据源中不断产生的，并且要求低时延的得到计算结果。在开源流式平台之前，要想实现流式计算，通常必须建立一个由队列和众多节点组成的网络来实现实时处理。节点处理队列消息，同时更新数据库，发送新消息给其它队列以供后续处理。然而，这种方法，大部份开发时间浪费在配置消息发送，部署 worker、部署中间队列，实时处理逻辑在代码中的比例相对较小；容错性差，要自己设计机制处理数据丢失和系统故障；伸缩性差，当单个节点的消息吞吐量太高时，需要重新增加服务器并且重新均衡负载。

主流的流式计算平台还有 Storm，S4，Pima，StreamBase 和 Borealis 等。Storm 是 Twitter 的流式开源平台，使用 Clojure 语言编写，具有可扩展性、容错性、易用性等特点，是当前最流行的流式平台。S4 是 Yahoo 的开源流式计算平台，使用 java 编写。特点是完全去中心化的对等结构，但是没有可靠处理的机制，编程语言也只支持 Java。Puma 是 Facebook 的流式计算系统，它是通过通过一些实时性改造，让原本针对批处理计算平台也可以进行实时计算。

表 3.1 流式平台对比

	Storm	S4	Puma
开发语言	Clojure	Java	Java
系统架构	主从式	对称	对称
资源利用率	高	低	低
恢复时间	长	长	短
状态持续	No	Yes	Yes

De-dup	No	No	Yes

3.3 Storm

Storm 解决了这些问题。Storm 是 Apache 旗下的开源项目。根据官方的定义，它是“分布式容错实时计算系统”。Storm 提供了流式计算的基本框架和容错、扩展的基本机制。Storm 让第三方人员专注于自己的业务需求，开发数据处理的逻辑和算法，而无需关心消息是如何传递的。

3.3.1 特点

简单的编程模型：Storm 公开一组实时计算原语，只需要简单几条语句，就可以写出复杂的数据流图

容错性：Storm 可以轻松管理工作进程，也可以应对节点的故障。

水平扩展：当计算任务增大时，我们可以谁是增加进程或服务器。

可靠的消息处理：Storm 保证每个消息至少能得到一次完整处理。任务失败时，它会负责从消息源重试消息，保证每个消息都被处理。

快速：每个节点每秒可以处理百万个元祖

本地模式：Storm 有一个“本地模式”，可以单机电脑上模拟集群的运行，非常有利于开发调试

编程语言无关性：Storm 是使用 Clojure 编写的，但是，Storm 的拓扑结构和 Bolt、Spout 几乎可以用当今市面上所有语言定义，包括 Ruby、PHP、Perl、Python、Javascript

3.3.2 架构

Storm 集群由一个主节点和多个工作节点组成。主节点的守护进程名为“Nimbus”，功能是进行代码分配、任务布置及故障检测。每个工作节点的守护进程名为“Supervisor”的，功能是监听工作节点的工作情况，开始并终止工作进程。

Nimbus 和 Supervisor 都是没有状态的，而且可以快速失败。这样使得它们就变得十分健壮。Nimbus 和 Superviosr 之间的协调是通过 ZooKeeper 来进行的。

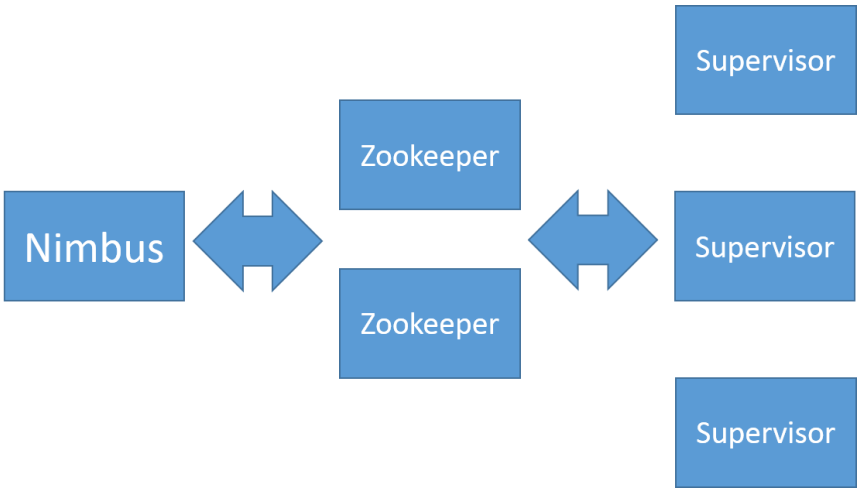


图 3.1 Storm 架构

Storm 在结构上也与 Hadoop 有很多相似之处。Nimbus：负责分配资源和调度任务，与 Hadoop 的 JobTracker 类似；Supervisor 接收 nimbus 分配的任务，类似 Hadoop 的 TaskTracker；Worker 中是具体分析的逻辑和进程，类似 Hadoop 的 Child。

表 3.2 Hadoop 与 Storm 对比

	Hadoop	Storm
系统角色	JobTracker	Nimbus
	TaskTracker	Supervisor
应用名称	Child	Worker
组件接口	Job	Topology

同时，我们也可以用 Storm 实现 Hadoop 的功能。

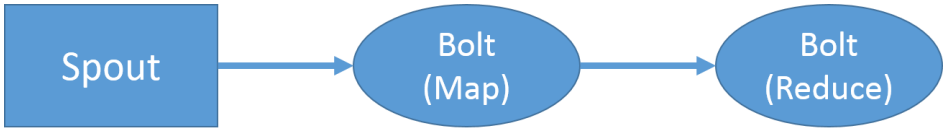


图 3.2 用 Storm 实现 Hadoop 功能

3.3.3 Topology

Topology 是 Storm 的基本逻辑单元。它是由多个数据流发射模块和数据流处理模块组成的节点网络。其最主要的主城部分是 Spout 和 Bolt。

Spout 是 topology 中产生源数据流的组件。通常情况下 spout 会接收数据源的数据，将其转换为 topology 内部运行的源数据。Spout 是一个主动的角色，其接口中有个函数 nextTuple()，storm 不停地调用此函数，监听数据源，用户只要在其中生成源数据即可。

Bolt 是 topology 中接受数据然后执行处理的组件。Bolt 可以执行过滤、函数操作、合并、写数据库等任何操作。Bolt 是一个被动的角色，其接口中有个 execute( )函数,在接受到消息后会调用此函数，用户可以在其中执行自己想要的操作。

Tuple 是一次消息传输的基本单位。序列 tuple 就组成了数据流 stream，一个 stream 是一个没有边界并且永不停止的 tuple 序列。

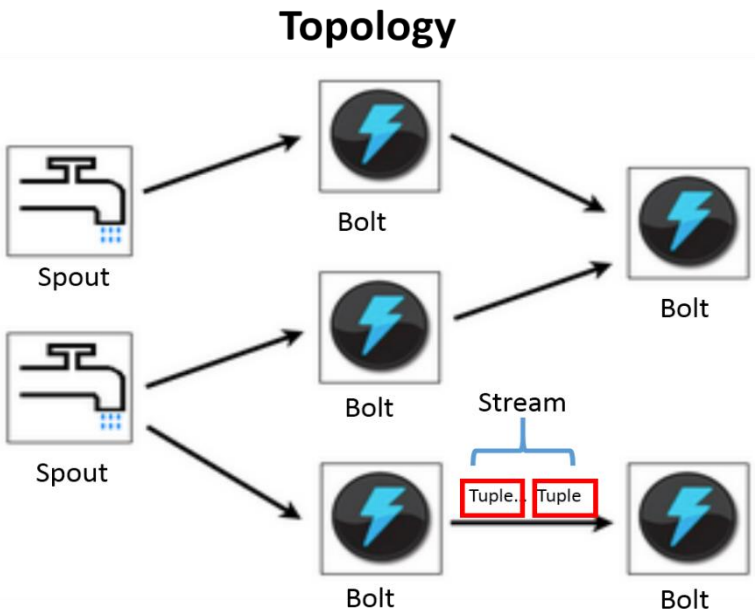


图 3.3 Topology 架构

3.3.4 Stream Groupings

在 Topology 中的每一个 Spout 和 Bolt 都可以设置并行度。对于每一个并行度，系统都会分配一个进程。那么数据在流动的过程中，到底选择哪一个进程，这就是 Stream Groupings。

Stream Grouping 定义了一个流在不同的 Bolt 任务间该如何被切分和传递。一共有有 6 个 Stream Grouping 类型：

随机分组：tuple 到 Bolt 的任务是随机分发的，每个任务都能保证得到相等数量的 tuple。

字段分组：分割数据流是通过指定字段进行的。相同字段的元组总是分发到同一个任务，不同字段的元组可能分发到不同的任务。

全部分组：tuple 被复制到 bolt 的所有任务

全局分组：所有流都被分配到 bolt 的同一个任务。也就是 ID 最小的那个 task。

无分组：这种情况下，你无需关心数据流是如何分组。等效于随机分组。

直接分组：是一个特殊的分组类型。元组生产者决定 tuple 由哪个元组处理者任务接收。

第4章　实时数据预处理与可视化

4.1　数据概览

本次拿到的数据是深圳市 2010 年-2012 年共三年的电压暂降记录，2 万余条记录，每条记录的特征主要有变电站、母线、相别、事件类型、开始时间、持续时间、持续电压。

表 4.1 电压暂降数据样例

变电站	母线	相别	事件类型	开始时间	持续时间(s)	持续电压(%)
110kV丹荷站	10kV 1M	B	暂降	2010-12-16 03:29:22	0.029	79.033
110kV丹荷站	10kV 1M	B	暂降	2010-12-16 03:29:23	0.029	79.290
110kV丹荷站	10kV 2AM	B	暂降	2010-12-16 03:29:23	0.029	79.077
110kV丹荷站	10kV 2AM	B	暂降	2010-12-16 03:29:24	0.029	79.283
110kV丹荷站	10kV 2BM	B	暂降	2010-12-16 03:29:25	0.029	79.247
110kV丹荷站	10kV 2BM	B	暂降	2010-12-16 03:29:26	0.039	79.225
110kV盐田站	10kV 2BM	A	暂升	2010-08-11 10:08:52	0.000	120.628
110kV盐田站	10kV 2BM	C	暂升	2010-08-11 10:08:57	0.016	123.350
110kV盐田站	10kV 2BM	C	暂升	2010-08-11 10:08:58	0.000	120.642
110kV盐田站	10kV 2BM	A	暂降	2010-08-11 10:09:00	0.094	41.500
110kV盐田站	10kV 2BM	B	暂升	2010-08-11 10:09:00	0.086	141.831

同时我们还有我们还有事故原因记录数据，变电站地理信息数据。

表 4.2 事故原因数据样例

时间	变电站	日志详情	故障原因
40179.32153	安良站	（龙岗局横岗所）安良站 F13 梧岗甲线接地切开关	大风
40179.76042	和平站	（宝安局福永所）和平站 F15 新三线跳闸	用户设备故障
40179.99028	红岭站	（罗湖局）红岭站 F15 桃园线跳闸	用户设备故障
40180.18542	南头站	（南山局）南头站 F12 中山线跳闸	用户设备被盗

40180.65278	盘古石站	（龙岗局坑梓所）盘古石站 F01 公园线跳闸	下雨
40180.91736	圳美站	（光明局）圳美站 F06 光侨线接地切开关	下雨
40180.92708	松岗站	（宝安局松岗所）松岗站 F4 西水线接地切开关（用户专线）	下雨
40181.29236	水田站	（宝安局石岩所）水田站 F57 北环线跳闸	用户电缆被盗
40181.78333	欢乐站	（南山局）欢乐站 F1 华联线跳闸	电缆故障
40181.92292	西乡站	（宝安局宝城所）西乡站 F45 河东二线跳闸	雨
40182.35833	田面站	（福田局）田面站 F51 华强二线接地跳闸	查无原因

深圳市有大量金融、电子、半导体、精密加工等高新技术、现代制造和现代服务企业入驻，大量敏感性电气设备的使用对电能质量提出了较高要求。同时，在拥有大量金融、数据中心、客户服务中心、医院、政府机关的地区，其对供电可靠性的高要求及出现电能质量问题时可能引起的巨大不良社会影响，使得这些供电区域或企事业单位对电能质量的要求极为严格。电能质量尤其是电压暂降问题日益引起高新企业和社会的关注，电能质量好坏经济和社会的进一步发展。换句话说，电能质量的好坏关乎到深圳市乃至广东省的产业升级能否顺利进行。表 4.1 展示了深圳市近年电压暂降事件汇总。

表 4.3 深圳市近年电压暂降次数

时间 等级	2011	2012	2013
10KV	616	1584	481
110KV	563	1440	1160
220KV	4	0	18

伴随着企业及重要用户对电能质量要求的不断提升，深圳地区电网电能质量尤其是电压暂降问题的监测及治理存在着不足之处。为了营造深圳市良好的经济和社会发展环境，吸引大量高新企业和现代化企业的入驻，保证企事业单位的优质用电，急需开展深圳配网的电能质量问题尤其是电压暂降的监测和风险评估，合理配置深圳配网的电能质量监测点，开展有效的电压暂降监测和风险评估，并开展对电压暂降干扰源的定位和查找工作。以深圳市电压暂降问题的监测、风险评估和干扰源定位为示范，对广东电网乃至南方电网电能质量问题的监测和风险评估等工作也将起到很好的带头作用和借鉴意义。

电网电能质量的提高会产生巨大的经济效益和社会效益，因此本研究具有相当的价值。

4.2 数据预处理

由于不同的数据流来自不同的数据源，具有不同的结构和组织形式，可能不正确、不完整或不一致。如果我们下一步想继续进行分析、挖掘和可视化的话，需要对它们进行数据预处理。

存在不完整的、含噪声的和不一致的数据是现实世界大型的数据库或数据仓库的共同特点。不完整数据的出现可能有多种原因。有些感兴趣的属性，如销售事务数据中顾客的信息，并非总是可用的。其他数据没有包含在内只是因为输入时认为是不重要的。相关数据没有记录可能是由于理解错误，或者因为设备故障。与其他记录不一致的数据可能已经删除。此外，记录历史或修改的数据可能被忽略。缺失的数据，特别是某些属性上缺少值的元组可能需要推导出来。

数据含噪声（具有不正确的属性值）可能有多种原因。收集数据的设备可能出故障；人或计算机的错误可能在数据输入时出现；数据传输中的错误也可能出现。这些可能是由于技术的限制，如用于数据传输同步的缓冲区大小的限制。不正确的数据也可能是由命名约定或所用的数据代码不一致，或输入字段（如日期）的格式不一致而导致的。重复元组也需要数据清理。

对于此次的暂降数据，我们需要进行时间格式的统一、变电站字符串的提取、变电站地理信息的填充、错误数据的删除等。

我们把来自两个数据源的数据——暂降数据和原因记录汇总，形成暂降故障原因记录表。暂降原因的匹配的主要依据是发生时间和地点。这个相当于将两个数据流 reduce 成一个数据流，起到聚合的作用。

Topology 中 Bolt 的设计如下：

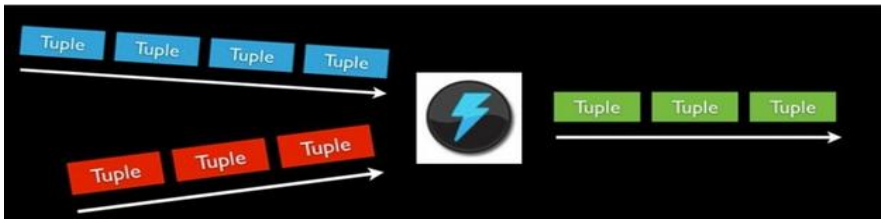


图 4.1 数据流聚合 Topology

汇总之后的数据，同一时间发生的三相暂降数据聚合在一起。同时，每个暂降数据都相应的都有原因记录。这样我们就可以进一步分析挖掘不同原因导致的暂降具有什么特点；每一次暂降是如何在电网中传播等。具有相当的价值。

Topology 设计如下：

TopologyBuilder builder = new TopologyBuilder();

builder.setSpout(“Spout_ElectricityQuality1”,new Reader());

builder.setSpout(“Spout_ElectricityQuality2”,new Reader());

builder.setBolt(“converge_Bolt”, newNormalizer())

.shuffleGrouping(“Spout_ElectricityQuality1").shuffleGrouping(“Spout_ElectricityQuality2");

表 4.4 和表 4.5 展示了融合后的数据。第一部分是发生时间和 ABC 三相的暂降时间和暂降深度；第二部分是故障原因的发生时间、原因描述和变电站的经纬度。

表 4.4 聚合后数据样例(第一部分)

变电站	发生时间	Atime	Adegree	Btime	Bdegree	Ctime	Cdegree
李朗站	2010-07-11 18:11:37	0.031	122.404	0.211	44.717	0.188	164.147
南澳站	2010-07-28 10:08:21	9.82	0.174	9.82	0.185	9.82	0.177
新百站	2010-07-28 10:08:23			0.344	61.921	0.352	55.432
李朗站	2010-08-02 07:56:33	0.109	197.013	0.156	15.963	0.234	175.808

东湖站	2010-11-13 10:55:30					0.492	16.655

续表 4.5 聚合后数据样例(第一部分)

变电站	发生时间	Atime	Adegree	Btime	Bdegree	Ctime	Cdegree
东湖站	2010-11-13 10:55:31			5.359	15.593		
简龙站	2011-03-23 10:12:34	9.859	0.574	9.859	0.406	9.852	0.501
简龙站	2011-04-17 16:10:01	0.102	43.415	0.148	172.472	0.047	143.793
简龙站	2011-04-17 16:10:02	0.125	43.341	0.156	179.668	0.07	146.96
简龙站	2011-04-17 16:10:03	0.086	47.801	0.109	155.897	0.055	139.456

表 4.5 聚合后的数据样例（第二部分）

变电站	纬度	经度	故障时间	故障描述
李朗站	22.5547815	113.9407789	2010-07-11 18:09:00	主电缆故障
南澳站	22.6884991	113.811898	2010-07-28 10:08:00	电缆故障
新百站	22.72905556	113.8650278	2010-07-28 10:08:00	电缆故障
李朗站	22.58380556	113.9380833	2010-08-02 07:54:00	用户自身改造破坏电缆
东湖站	22.58898619	113.9206087	2010-11-13 10:53:00	施工挖断电缆
东湖站	22.57866667	113.9243611	2010-11-13 10:53:00	施工挖断电缆
简龙站	22.59540406	113.8788432	2011-03-23 10:11:00	主电缆被施工挖断
简龙站	22.59540406	113.8788432	2011-03-23 10:11:00	主电缆被施工挖断
简龙站	22.72905556	113.8650278	2011-04-17 16:10:00	电缆故障

4.3 实时电压暂降可视化

视觉信息是人类最主要的信息来源。研究结果表明：人类日常生活中所接受的信息约 80%来自于视觉信息。基于人类强大的视觉潜能，二十世纪八十年代后期，一项新的技术——可视化技术（Visualization Technology）被提出并随后得到迅速发展。可视化是利用计算机图形学和图像处理技术，将数据转换成图

形或图像在屏幕上显示出来并进行交互处理的理论、方法和技术。其涉及到计算机图形学、图像处理、计算机视觉、计算机辅助设计等多个领域，成为研究数据表示、数据处理、决策分析等一系列问题的综合技术。

科学计算可视化（Visualization in Scientific Computing）是 1987 年被发达国家提出后逐渐发展起来的一个新的研究领域。通过科学计算可视化来启发和促进对自然规律的更深层认识，从而发现规律并应用于生产领域。科学计算可视化是指应用计算机图形学和图像处理技术，将在科学计算过程中产生的数据和计算结果转换为图形或图像在屏幕上显示出来，并进行交互处理的理论、方法和技术[1]。实际上，随着技术的不断进步，科学计算可视化的含义已经大大扩展，不仅包括科学计算数据的可视化，还包括工程计算数据的可视化以及测量数据的可视化等。科学计算可视化是覆盖多门学科的研究领域，融合了计算机图形学、图像处理学、科学与符号计算、计算机视觉等领域的知识。

数据可视化（Data Visualization）是关于数据之视觉表现形式的研究。其中，这种数据的视觉表现形式被定义为一种以某种概要形式抽提出来的信息，包括相应信息单位的各种属性和变量。数据可视化技术的基本思想是将数据库中每一个数据项作为单个图元元素表示，大量的数据集构成数据图像，同时将数据的各个属性值以多维数据的形式表示，可以从不同的维度观察数据，从而对数据进行更深入的观察和分析。

数据可视化技术包含以下几个基本概念：

(1) 数据空间 (Data Space)：也称作多维数据空间，是由多维属性和多个元素组成的数据集构成的多维空间。

(2) 映射空间 (Mapping Space)：也称作投影空间，是将多维数据按照一定的函数或规则转换后得到的低维可视空间。

(3) 多维数据分析 (Multidimensional Data Analysis)：是指对多维数据进行切片、切块、旋转等动作剖析数据，从而能多角度多侧面地观察数据。

(4) 多维数据探索 (Multidimensional Data Exploration)：是指利用一定的算法和工具对多维数据蕴涵的信息进行搜索，得到有用、新颖的信息。

(5) 多维数据可视化 (Multidimensional Data Visualization)：是指将大型数据集中的数据以图形或图像形式表示，并利用数据分析和挖掘工具开发其中未知信息的处理过程。

数据可视化技术主要有以下三个特点：

(1) 交互性：用户可以方便地以交互的方式管理和处理数据。

(2) 多维性：应用对象或事件的数据具有多维属性和变量，而数据可以按其每一维的值对其进行分类、排序、组合和显示。

(3) 可视性：数据可以通过图象、曲线、二维图形、三维体和动画等形式来显示，并可对其模式和相互关系进行可视化分析。

在实现数据可视化之前，首先要进行数据的预处理，这个我们在第四章中已经完成。

每次电压暂降都是分布在 ABC 三相上，都具有不同的暂降深度和持续时间。同时，一些原因造成的暂态事故可能引起电网大规模的波动，造成经济损失。我们如果可以将每一次暂降根据地理位置可视化，就可以实时动态显示出暂降事件的分布和每一个暂降事件的传播。这对未来电网的规划具有重要意义。

我们把上面得到的暂降事故原因记录表通过一个 Spout 作为数据源，利用一个 Bolt 得出能量函数。输出的数据流利用 Python 持续调用微软地图的接口，实时动态显示暂降事件的分布。

我把三年的暂降数据依据持续时间、暂降深度和能量可视化分别做成视频，下面是三个视频的截图。

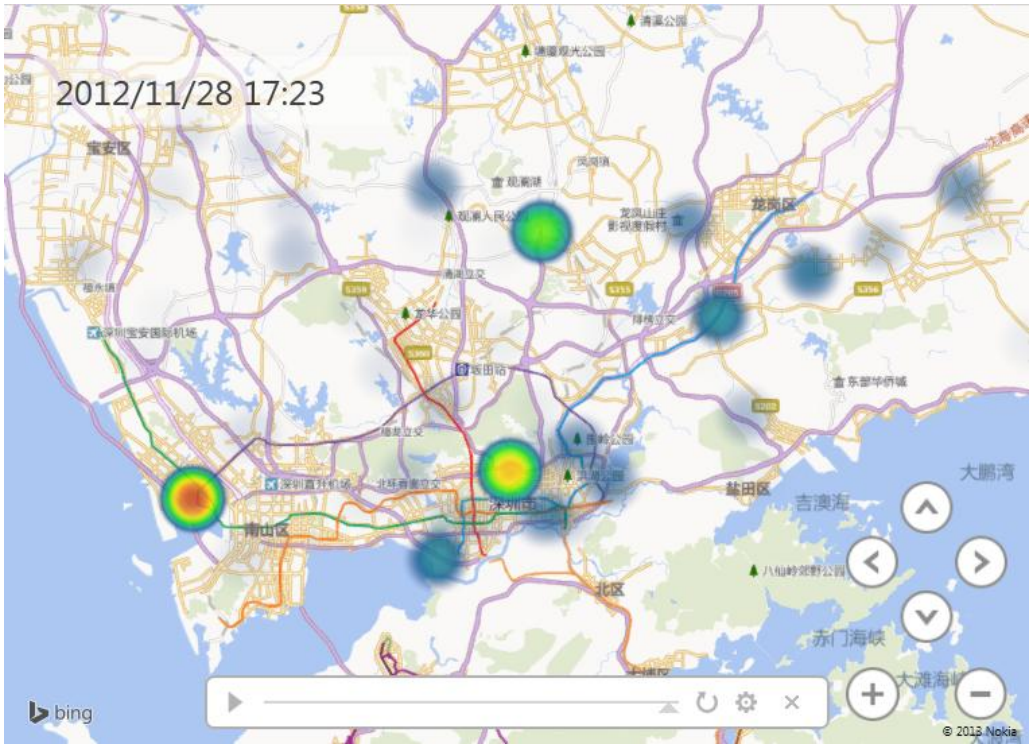


图 4.2 持续时间可视化

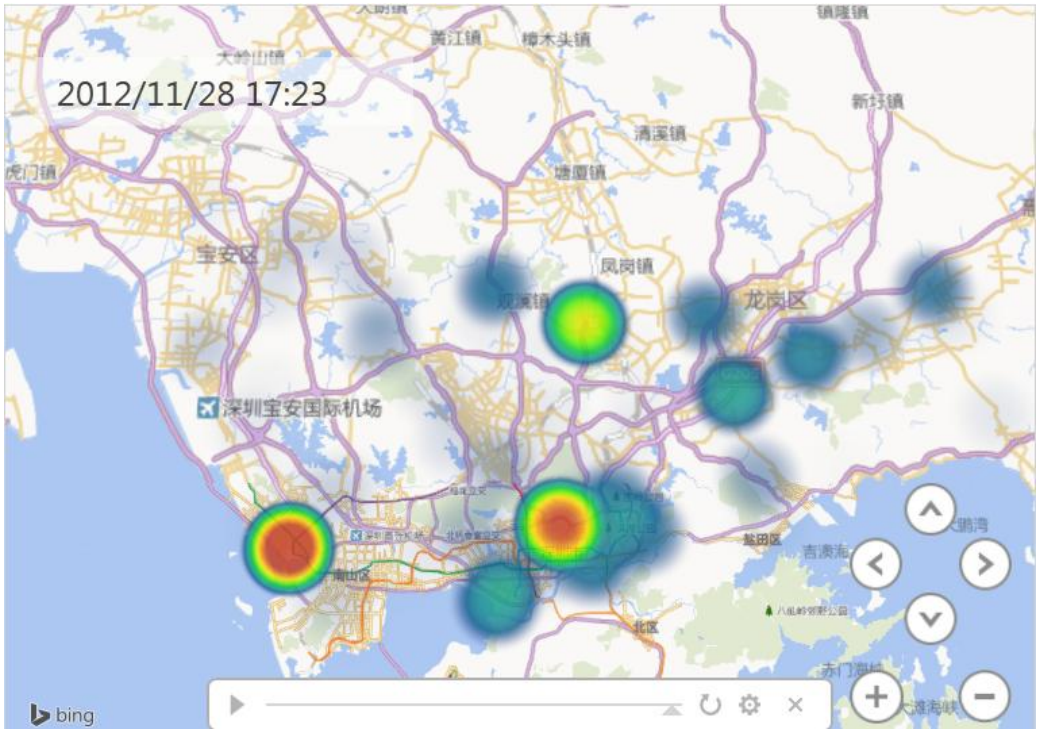


图 4.3 暂降深度可视化

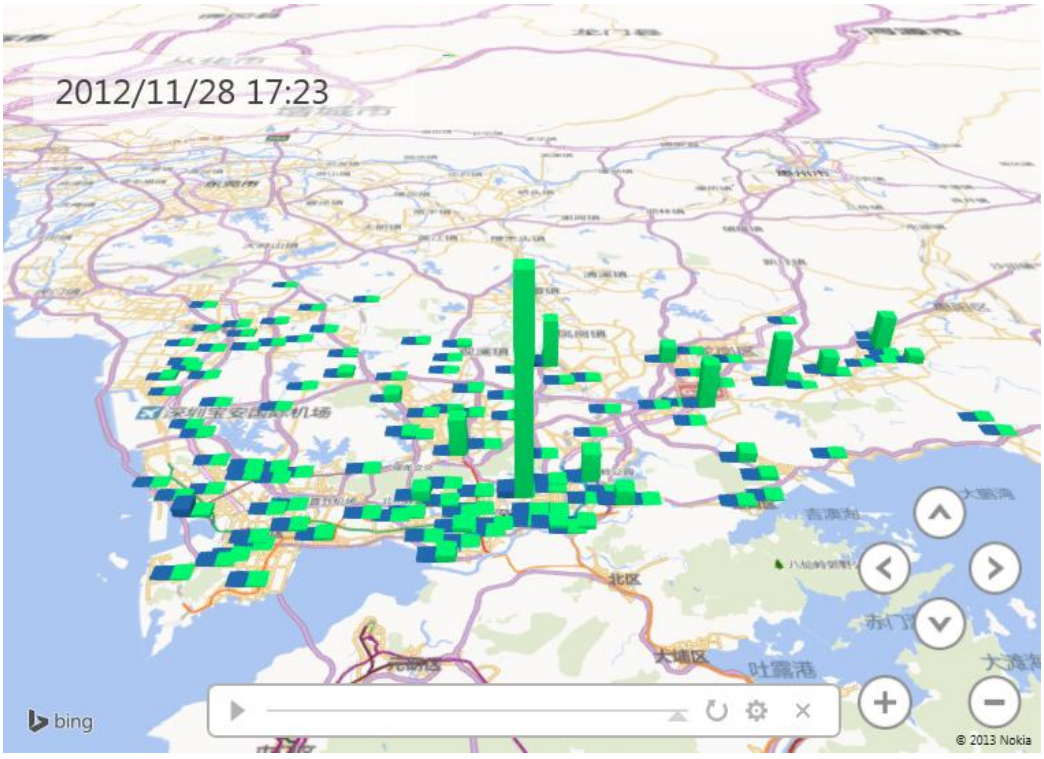


图 4.4 暂降深度和持续时间可视化

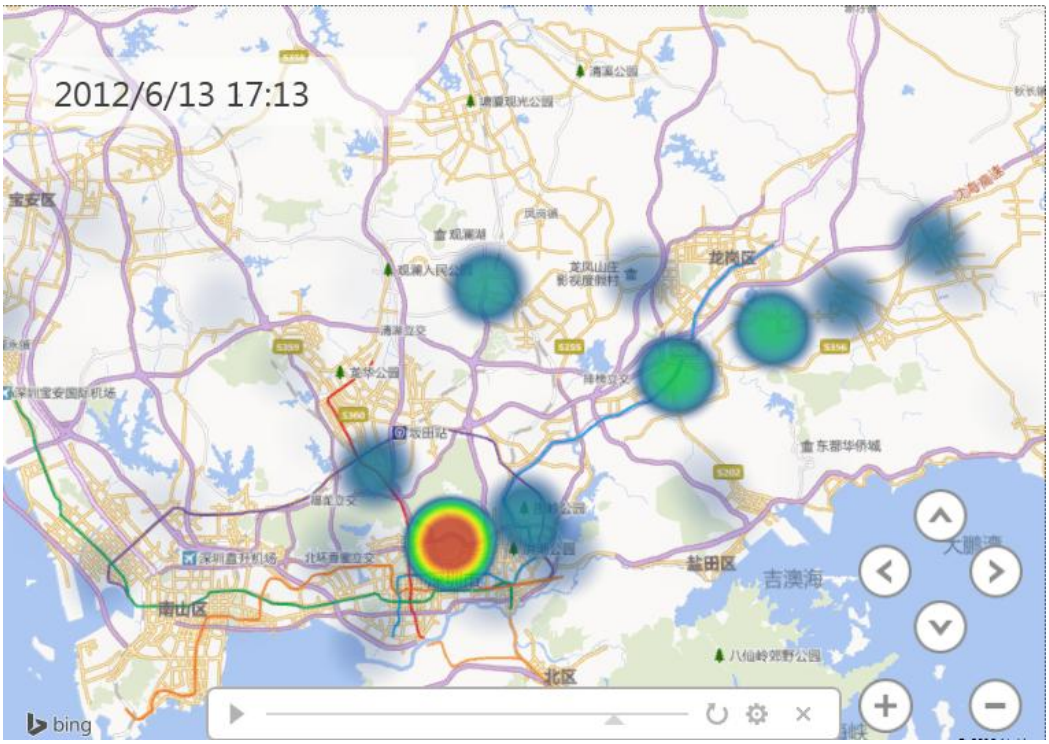


图 4.5 暂降能量可视化

第5章　实时电力数据压缩研究

5.1　数据压缩的意义

电压暂降事件中，有一部分是外部引起的，还有一部分是由于一个节点的电压暂降通过电网传播，最终影响其他节点。在深圳市三年工 8891 条电压暂降记录中，只有 3160 条是独立的源事件，占 35.5%；其余 64.5%全部是被动的、通过电网传播引起的。可以发现，由于电网波动引起的电压暂降事件与源事件在持续时间和暂降深度上具有很高的相关性。下面是 2010 年 4 月 22 日，由田寮站引起的系列电压暂降事件。我们可以看出 6 条记录的持续时间和暂降深度具有很高的相似性。

表 5.1 田寮站系列电压暂降事件

站名	母线	发生时间	C 相持续时间	C 相深度
田寮站	110kV 1M	2010-04-22　18:01:01	0.313	75.508
公明站	110kV 1M	2010-04-22　18:01:02	0.305	75.983
田寮站	110kV 1M	2010-04-22　18:01:03	0.047	76.239
公明站	110kV 1M	2010-04-22　18:01:04	0.047	76.731
上村站	110kV 1M	2010-04-22　18:01:06	0.344	69.437
上村站	110kV 1M	2010-04-22　18:01:08	0.078	70.149

同时，A、B、C 三相之间的数据也有着很高的相似性。这次的数据中，有 21%是只有一相发生电压暂降事件；其余 79%都是两相或三相发生。下表中的数据描绘了同一事件不同相别之间的相关性。

表 5.2 ABC 三相之间的相关性

福永站	0.305	175.173	0.313	41.928	0.328	39.368
清水河站	9.813	0.966	9.813	0.97	9.813	1.109
大水坑站	0.078	19.822	0.078	16.092	0.078	15.884
水田站	3.078	0.091	3.07	0.097	3.078	0.094

育新站	0.078	19.589	0.078	15.983	0.07	15.695
塘尾站	0.078	19.902	0.078	16.066	0.07	15.852

如果我们可以对不同事件，或同一事件不同相别之间的数据进行压缩，可以大大减小电网数据中心的存储压力、降低成本，同时也可以减少计算和分析的成本。

5.2 主成分分析法

首先我们采用主成分分析（PCA）方法。主成分分析的核心思想是：当多个变量近似线性相关是，我们将其投影到方差最大的方向上，从而选出最重要的变量。

利用主成分分析进行数据压缩的流程是：

1) 对样本数据进行归一化 $\frac{x_i(t)-e(x_i)}{\sqrt{\delta^2(x_i)}}$
2) 计算归一化样板数据协方差矩阵 XX^T
3) 计算 XX^T 的特征值和对应的特征向量
4) 对特征向量进行排序，选取前 m 个特征值所对应的特征向量 $q(1)q(2),q(m)$作为主成分的方向
5) 计算每个样本在主成分方向上的投影

$$y(t)=\left[q(1),q(2),\cdots q(m)\right]^T x(t)$$

这也是压缩后的需要存储的数据。

6) 如果需要进行数据的还原。则还原后的数据为

$$\tilde{x}(t)=[q(1),q(2),\cdots,q(m)]y(t)$$

为了更好地还原不同相别之间的差异，我们再来尝试概率主成分分析（PPCA），它与传统的主成分分析(PCA) 相比，克服了简单的“舍去”其它非主成分因子，能进一步提取出更好地描述样本数据的特征，为后续提高识别性能奠定了基础。PPCA 将“舍去”的特征因子作为噪声成分进行估计，并认为该噪声成分符合高斯模型，结合最大似然的分析方法，估计出模型的参数，得到样本的特征空间克服了简单的丢弃 其他非主成分因子。

对每个数据样本 t 是 N 个 d 维向量，并假设存在 q 维（ $q<d$ ）的隐形向量 x 与观测满足

$$t=Wx+\mu+\varepsilon$$

其中

$d \times q$ 的因子载荷矩阵

μ 为样本均值

ε 为噪声

隐形变量服从高斯分布 $x \sim N(0, I)$

$\mu = \frac{1}{N}\sum_{i}^{N} x_i$ ， $\varepsilon \sim N(0, \psi)$ 为对角协方差矩阵。

根据一系列推导，PPCA 的最大似然估计为

$$\sigma^2 = \frac{1}{d-q}\sum_{j=q+1}^{d} \lambda_j$$
$$W = U_q(\Lambda_q - \sigma^2 I)^{1/2} I$$

其中

$\lambda_i (i = q+1, q+2, \cdots d)$ 为样本协方差矩阵 $S = \frac{1}{N}\sum_{i=1}^{N}(x_i - u)(x_i - u)^T$ 的最小的 $d-q$ 个特征向量。

$\Lambda_q = diag(\lambda_{q+1}, \lambda_{q+2}, \cdots \lambda_d)$

λ_k 对应的特征向量为 U_q 的第 k 个列向量

I 为 $d-q$ 维的单位矩阵

压缩后的数据 X

$x = W^T(t - \mu)$

恢复的数据 $\hat{t}$

$\hat{t} = Wx + \mu$

5.3 实验结果

下面我们利用主成分分析法对 ABC 三相的数据进行压缩。

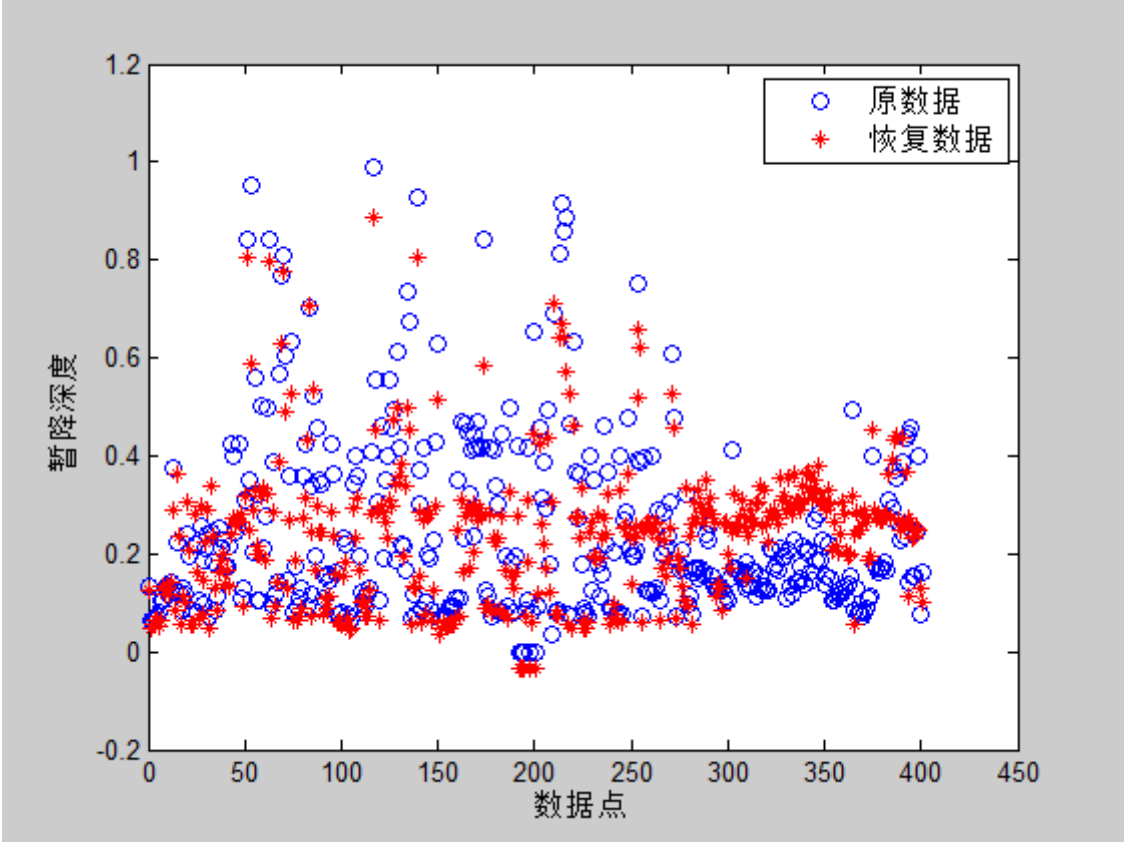


图 5.1 PCA 压缩数据与原数据对比（一个主成分）

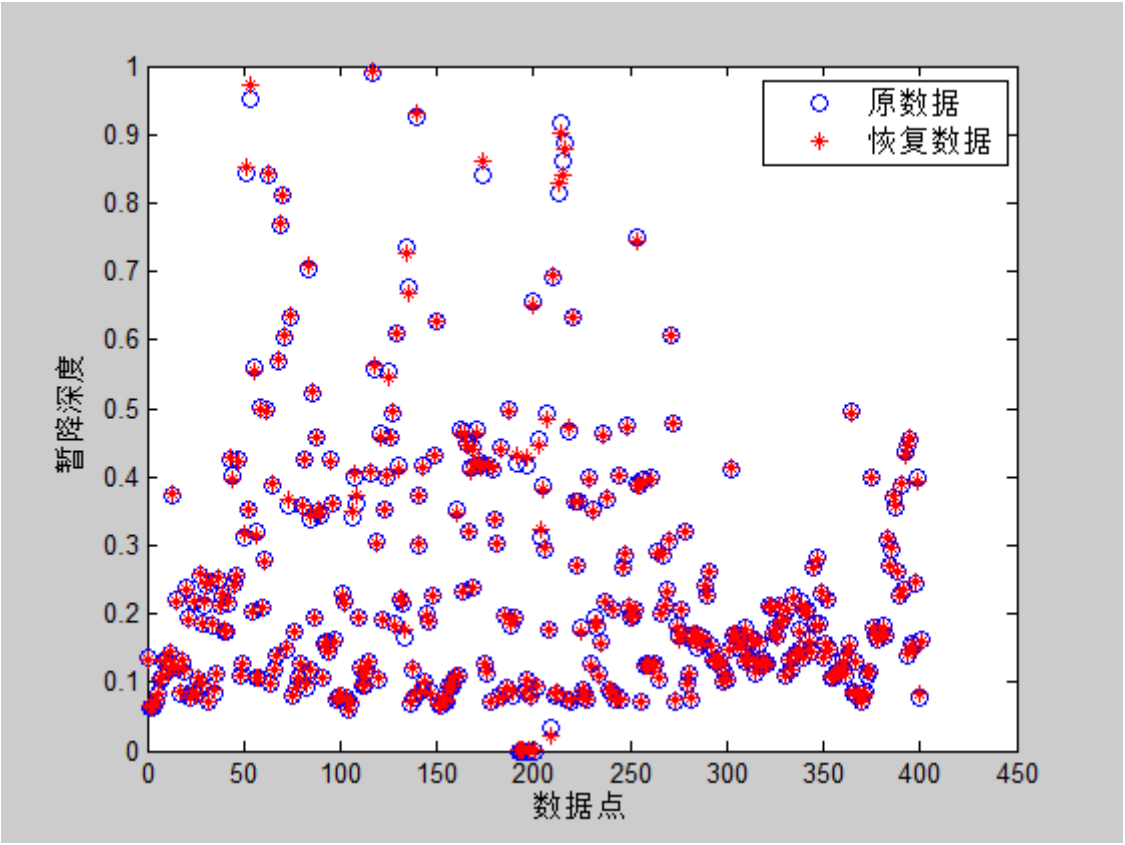


图 5.2 PCA 压缩数据与原数据对比（两个主成分）

下面是将暂降持续时间数据分别提取两个和一个主成分的效果图。其中，蓝色是原数据，红色是恢复后的数据。

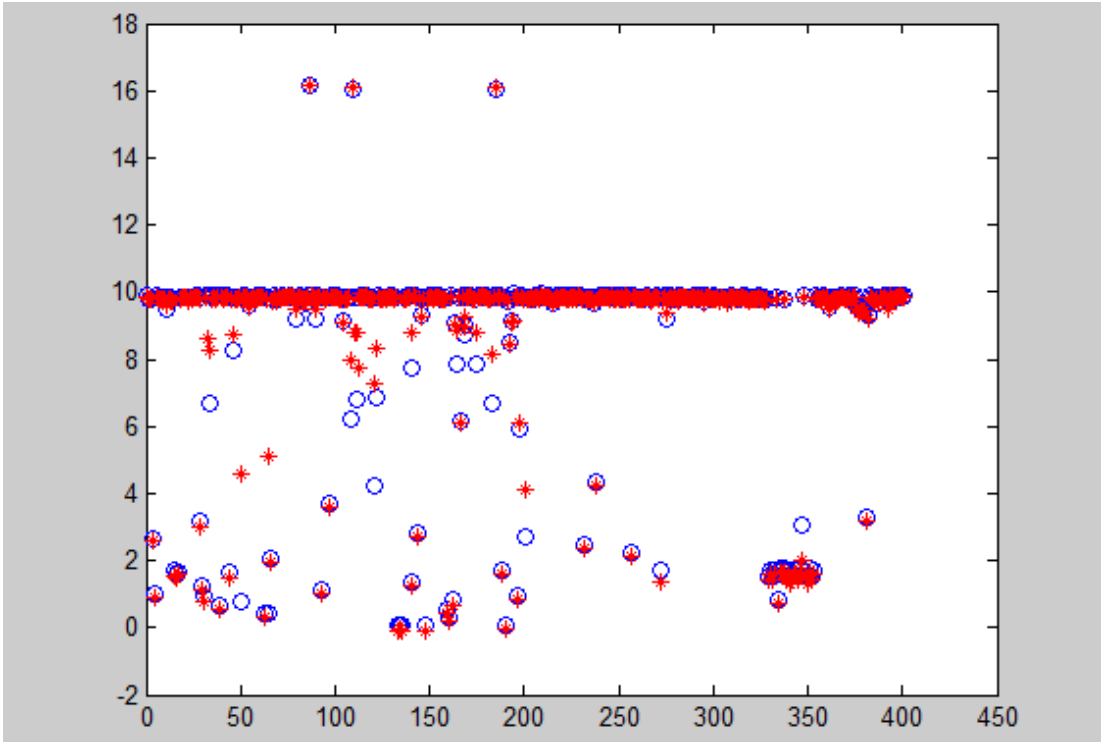


图 5.3 持续时间数据恢复后与原数据对比（一个主成分）

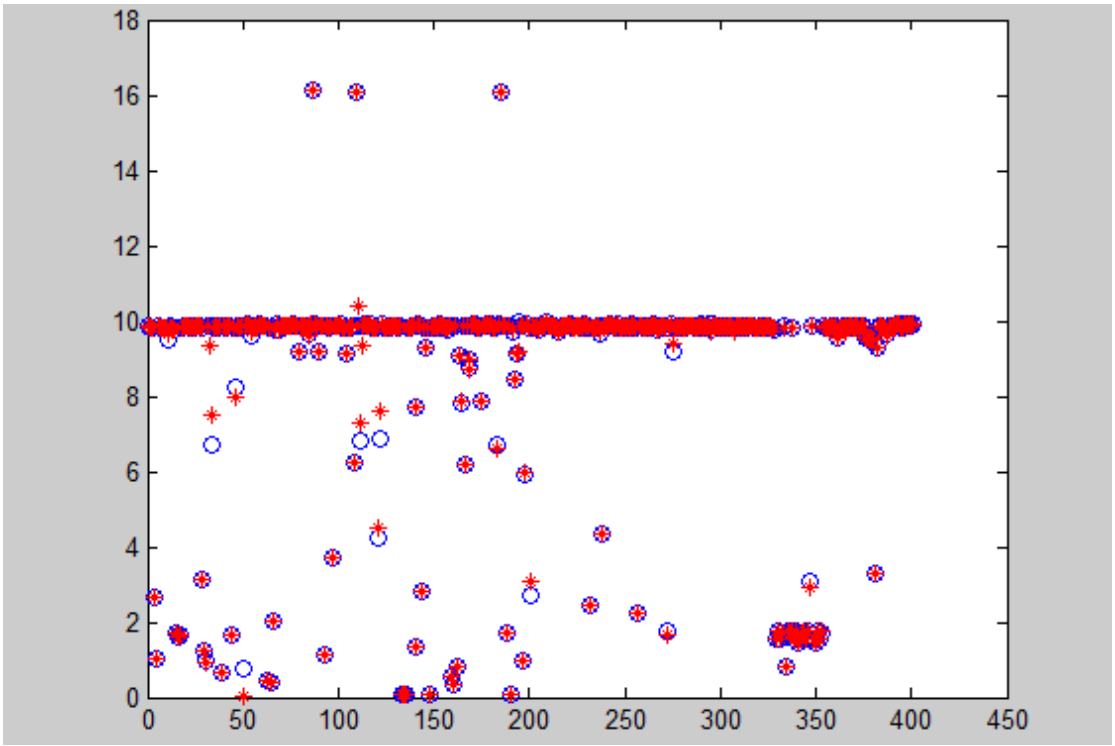


图 5.4 持续时间数据恢复后与原数据对比（两个主成分）

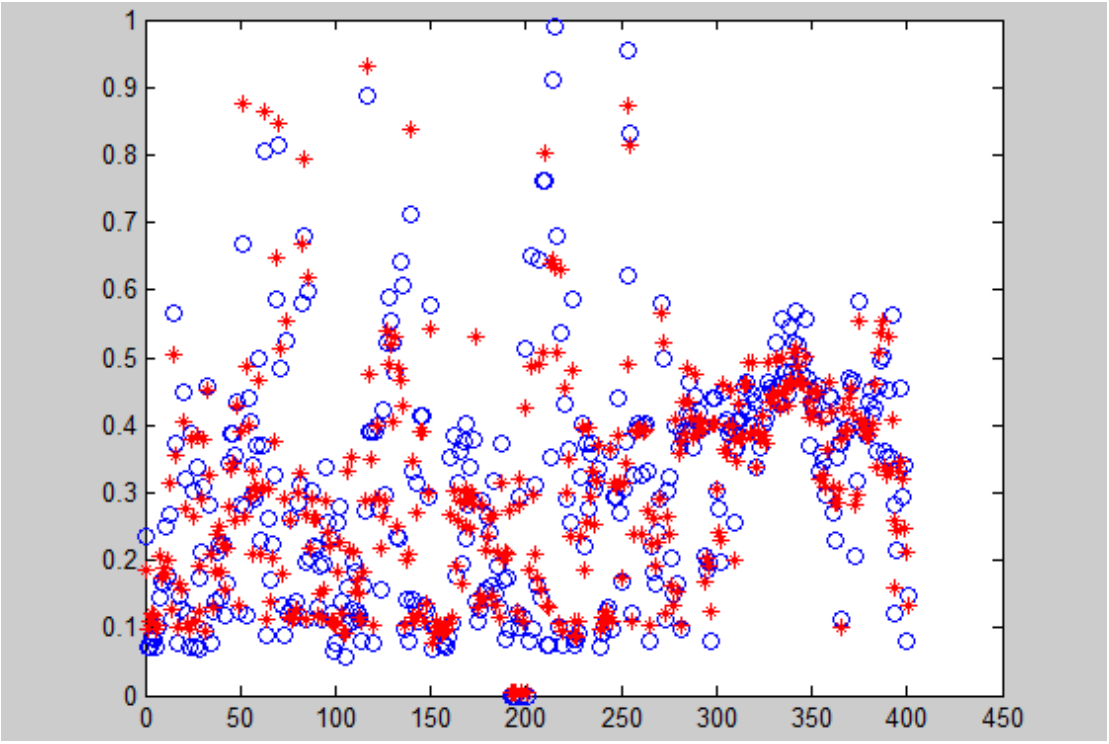


图 5.5 暂降深度数据恢复后与原数据对比（两个主成分）

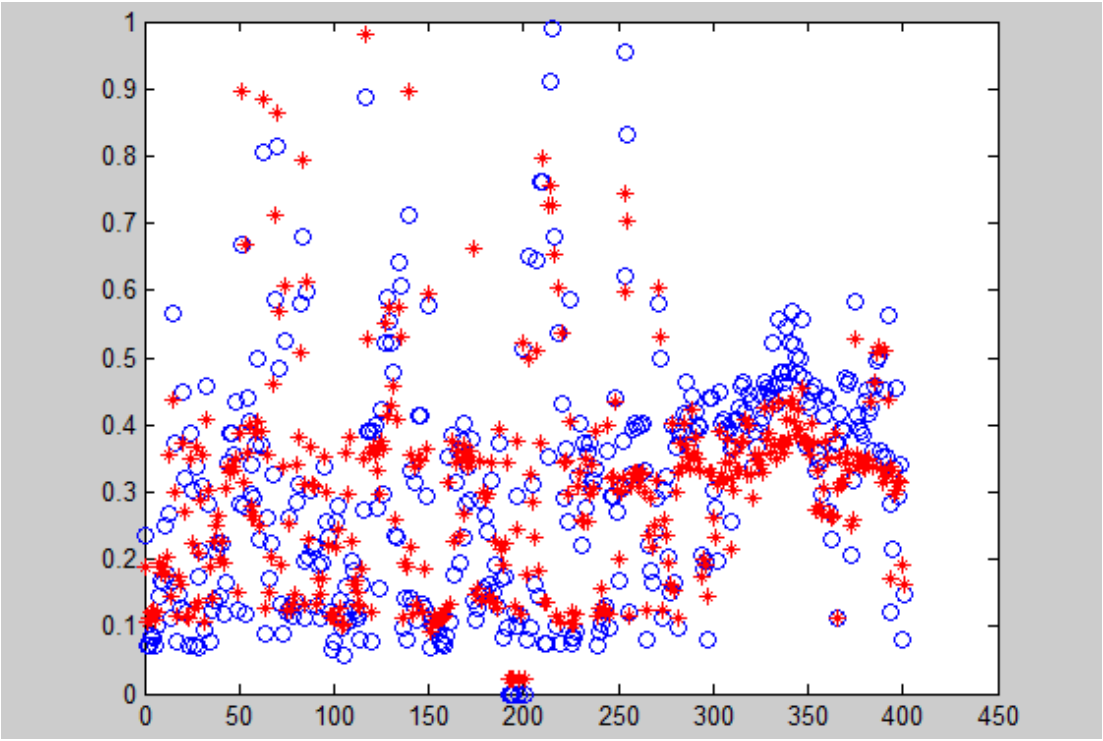


图 5.6 暂降深度数据恢复后与原数据对比（一个主成分）

从上面我们可以看出，对于暂降持续时间，当用两个主成分时，恢复出的数据与原数据已经非常相似，证明三相的持续时间具有很强的相关性。

恢复的暂降深度数据基本可以反映原数据，但是具有一定的误差。

5.4 压缩比与误差

数据压缩最重要的两个特征是压缩比和误差。压缩比指压缩前与压缩后的文件所占的磁盘空间比值，也就是就是经过压缩后，新数据的大小是原来的几分之一；误差是经过压缩后再恢复的数据于原数据的差别。

我们首先对同一暂降事件不同相别的数据进行数据压缩。首先，要抽取出具有三相数据记录的暂降事件，工 405 条。由于每条数据是 A、B、C 三相记录，所以我们抽取两个主成分或一个主成分。

数据压缩最重要的两个特征是压缩比和误差。压缩比指压缩前与压缩后的文件所占的磁盘空间比值，也就是就是经过压缩后，新数据的大小是原来的几分之一；误差是经过压缩后再恢复的数据于原数据的差别。

PCA 的误差：

$$error_{PPCA} = \frac{\sum_{t=1}^{N}\left(\tilde{x}(t)-\hat{x}(t)\right)^{T}\left(\tilde{x}(t)-\hat{x}(t)\right)}{\sum_{t=1}^{N}\tilde{x}^{T}(t)\tilde{x}(t)}$$

下表是抽取两个主成分和一个主成分的误差。可以看到，主成分数越大，误差越小，这证明保存了更多的数据信息。同时压缩误差最大只有 0.08，这对于电网数据是非常令人满意的。

表 5.3 持续时间数据恢复后与原数据误差

主成分个数	1	2
误差	0.016	0.00021755

表 5.4 暂降深度数据恢复后与原数据误差

主成分个数	1	2

误差	0.0815	0.0287

再来看压缩比压缩比 CR：

假设样本数据是 n 维向量，共有 N 个；我们提取出了 m 个主成分，则我们要存储一下这些数据：

在 m 个主成分方向上的投影，共 $m \times N$

m 个主成分向量，共 $m \times n$

样本均值，共 n 个

所以，压缩比为：

$$CR = \frac{m \times N + m \times n + n}{n \times N}$$

当提取两个主成分的时候： $CR = \frac{405 \times 2 + 2 \times 3 + 3}{405 \times 3} = 0.674$

当提取一个主成分时： $CR = \frac{405 \times 1 + 1 \times 3 + 3}{405 \times 3} = 0.338$

表 5.5 压缩比

	1	2
压缩比	0.338	0.674

对于三相都发生暂降的数据，当我们抽取一个主成分的时候，数据量只有原来的 0.33 倍。这可以大大减少电力数据中心的存储量，降低成本。

第6章 电压暂降事件原因分析

找出电压暂降事件的原因对于事后维修和提前预防具有重要的意义。总体来说，引起电压暂降的根本原因主要有电流突然增加，大电流在流过阻抗时使得电压下降。引起电压暂降的事故可能是企业内部，比如大型感应电动机的突然启动；也可能来自供电部门，比如重合闸；或是外力因素导致的，如雷击、施工破坏电缆等。不同的原因对应的暂降时间具有不同的特征。比如，变压器每一项的饱和程度不一样，因此引起的暂降也是不对称的；而电动机引起的电压暂降是对称的。

深圳市的故障操作是通过工程师手动记录的，尤其在原因描述方面，具有非结构化的特点。典型的电压暂降原因有：雷雨、用户设备故障、电缆故障（包括主电缆故障、支线电缆故障、用户电缆故障、电缆中间头故障）、开关故障、外力引起（包括动植物、施工挖断、电缆被盗等）。这其中，雷雨和电缆故障是分布最多的两个因素。在这章中，我们首先利用 Kmeans 聚类和 Apriori 关联算法分析、雷雨引起的电压暂降事件的特征和电缆故障导致的事件特征。然后我们利用支持向量机对两种事件进行分类，对未来数据原因的判断提出依据。

6.1 雷雨导致的暂降事件特征分析

我们的数据源是电压暂降事件记录表、和电压暂降原因记录表。我们首先通过 4.1 中的流聚合把二者合二为一，形成电压暂降原因记录表。我们以 180s（3 分钟）为时间匹配的依据，共匹配生 502 条结果。在这 502 条结果中与雷雨相关的原因结果共有 145 条。

然后，我们分别画出雷雨导致电压暂降事件的幅值——时间分布图（ITI 图）。期中横轴为持续时间的对数，纵轴是幅值。

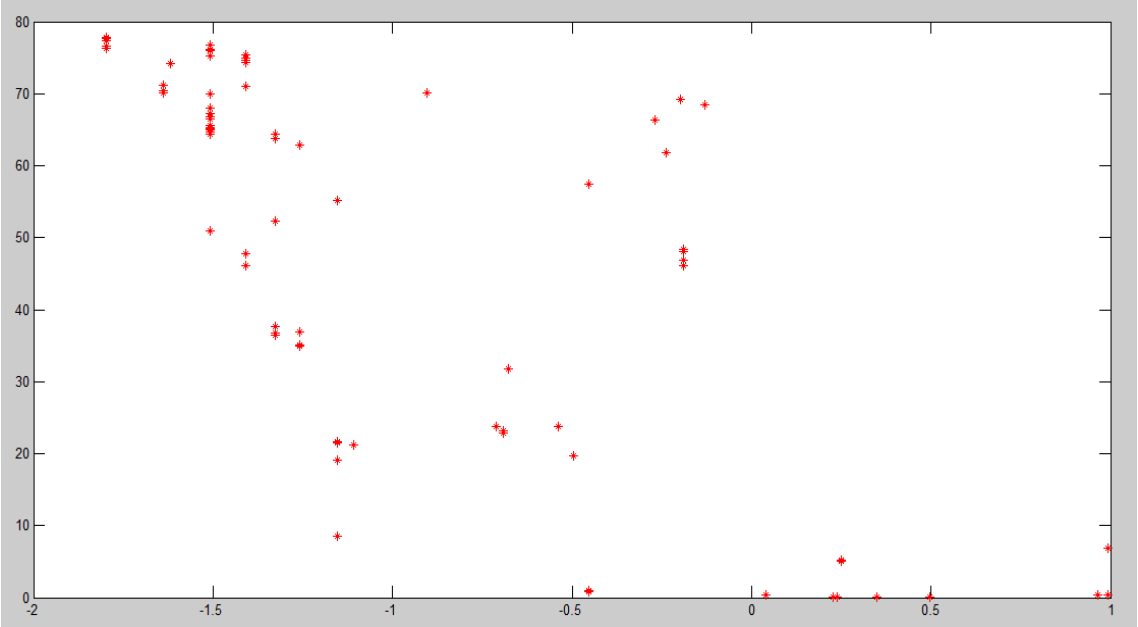


图 6.1 雷雨导致的电压暂降事件 A 相 ITI 图

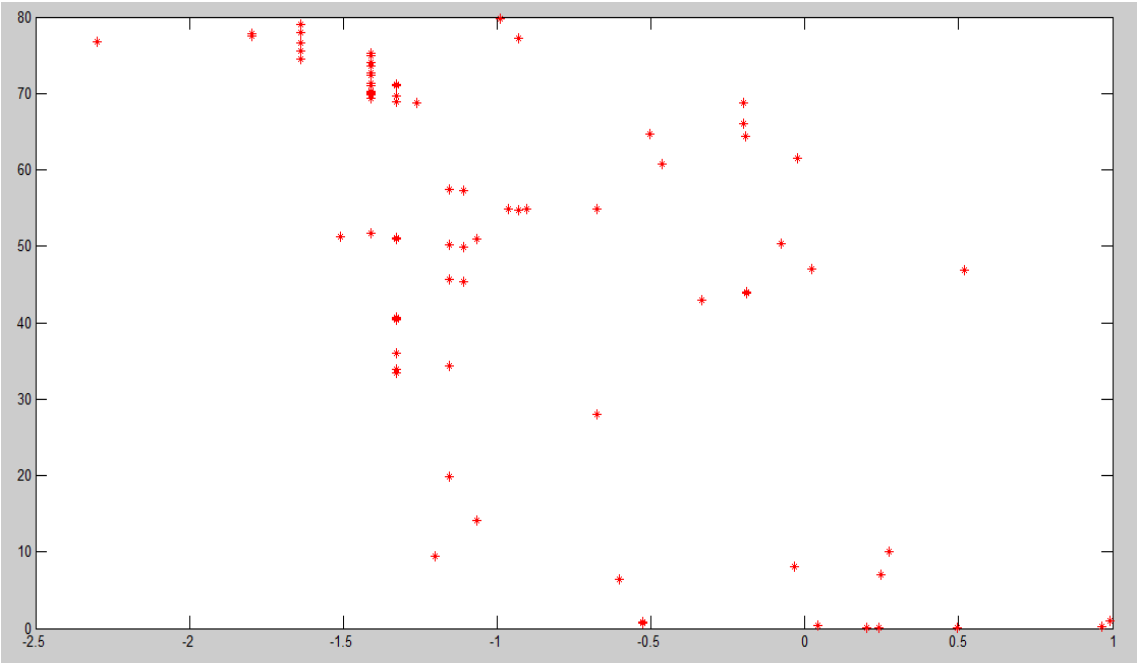


图 6.2 雷雨导致的电压暂降事件 B 相 ITI 图

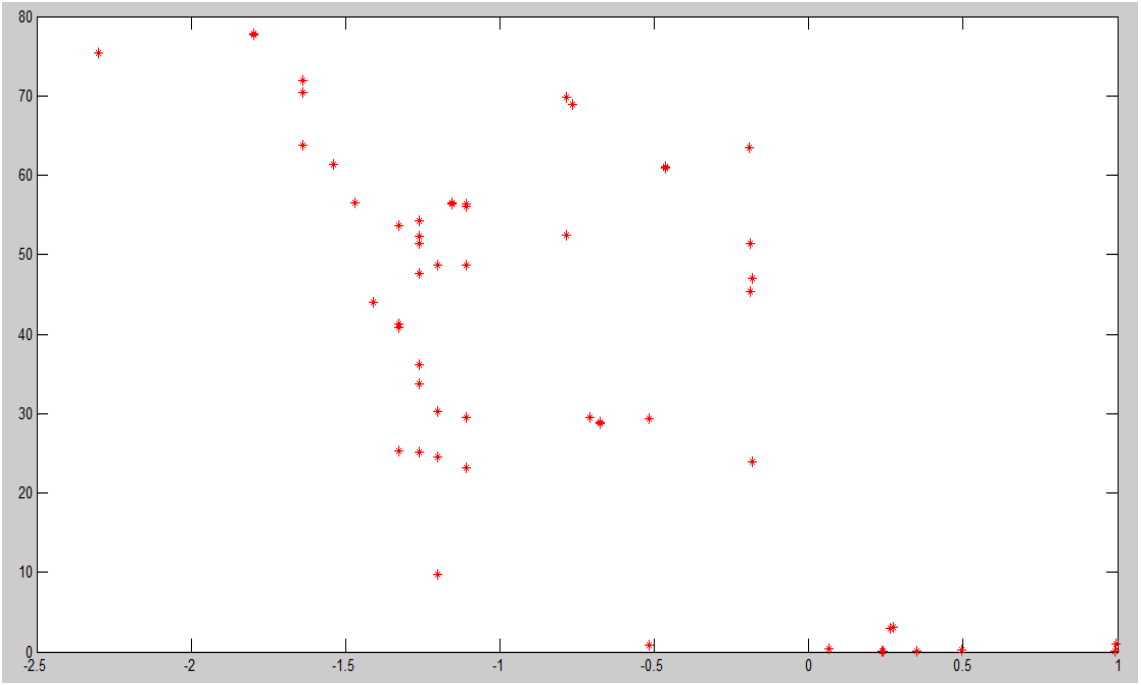


图 6.3 雷雨导致的电压暂降事件 B 相 ITI 图

通过这三张图我们可以发现，雷雨导致的电压暂降事件的暂降深度普遍比较低：大多集中在 60%-80%之间；持续时间短（0.5s 以内）。

为了更清楚的对比不同原因导致的电压暂降事件的特点，我们用红色点来描绘雷雨导致的电压暂降事件，蓝色点来描绘其余所有的电压暂降事件。

我们可以看到，其他原因导致的电压暂降事件整体分布比较均匀，期中还有一部分期中分布在高深度长时间的区域内。由于当电压暂降在 10s 以上时会触发电网的保护装置，因此有一大部分集中在右下角。

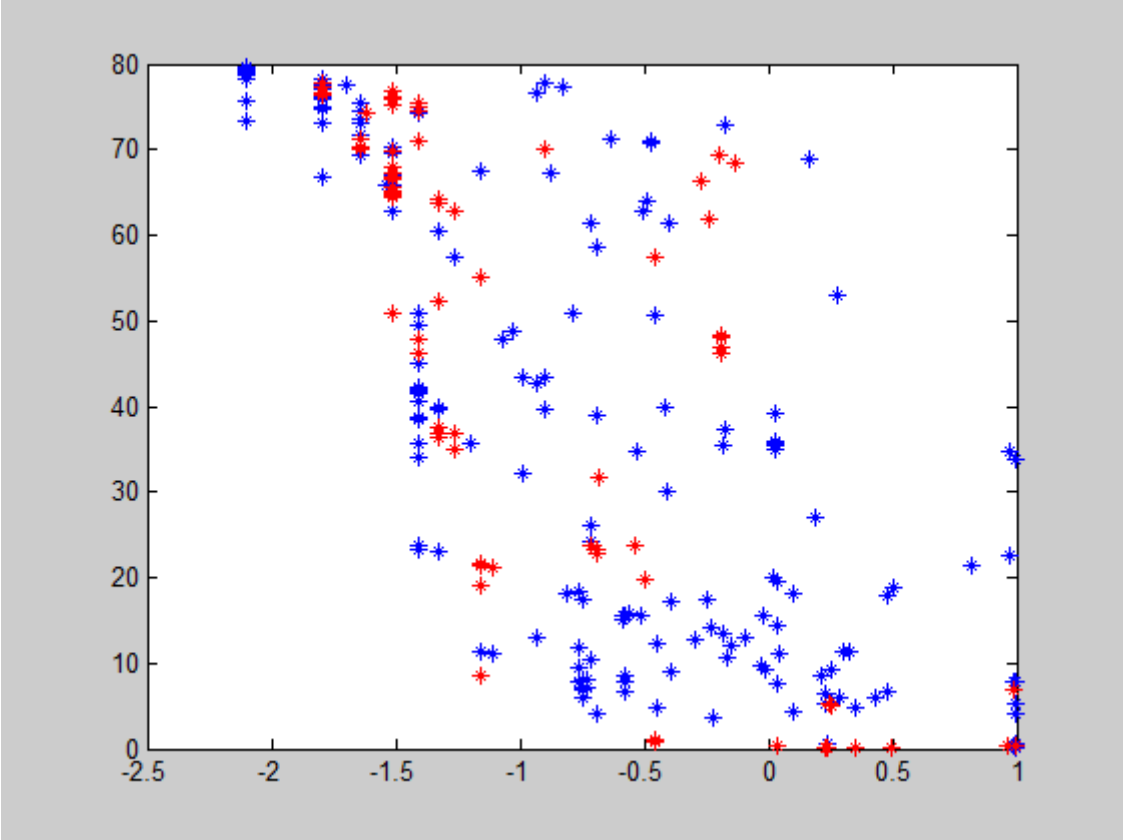


图 6.4 雷雨导致的电压暂降事件与其他原因对比

为了尽一份得出雷雨导致电压暂降事件的特点，我们将 A、B、C 三相的事件汇总。然后利用 K-means 算法进行聚类。K-means 算法的基本思想是：以空间中 k 个点为中心进行聚类，对最靠近他们的对象归类。通过迭代的方法，逐次更新各聚类中心的值，直至得到最好的聚类结果。

假设要把样本集分为 c 个类别，算法描述如下：

（1）适当选择 c 个类的初始中心；

（2）在第 k 次迭代中，对任意一个样本，求其到 c 各中心的距离，将该样本归到距离最短的中心所在的类；

（3）利用均值等方法更新该类的中心值；

（4）对于所有的 c 个聚类中心，如果利用（2）（3）的迭代法更新后，值保持不变，则迭代结束，否则继续迭代。

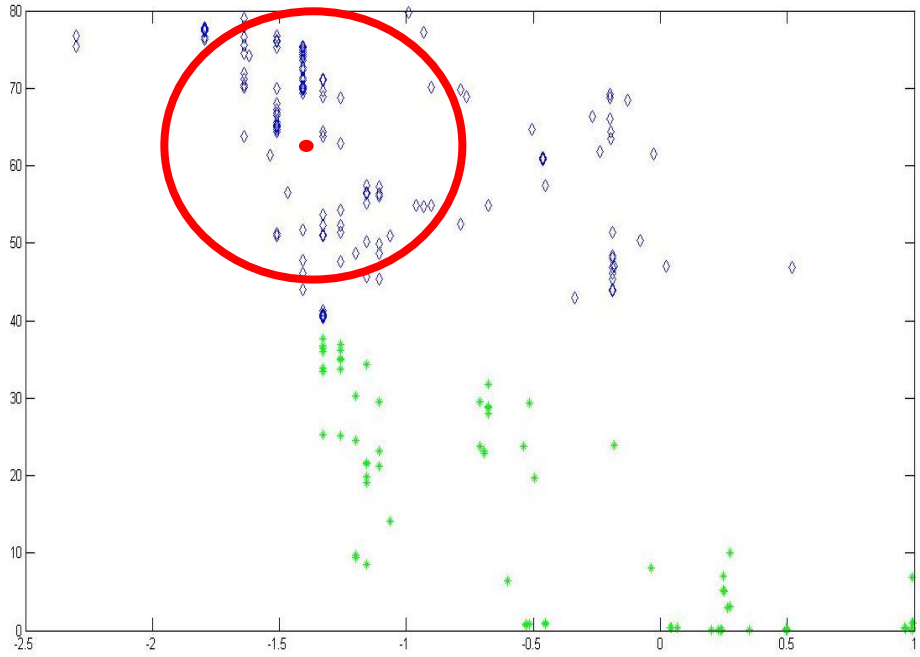


图 6.5 雷雨导致的电压暂降事件与其他原因对比

另外，我们还可以通过 Apriori 算法，利用关联规则来挖掘雷雨导致的电压暂降事件的特点。1994 年 Agrawal 提出的 Apriori 算法是挖掘完全频繁项集中最具有影响力的算法。算法有两个关键的步骤：一是发现所有的频繁项集；二是生成强关联规则。发现频繁项集是关联规则挖掘中的关键步骤。在 Apriori 算法中还利用了“频繁项集的子集是频繁项集，非频繁项集的超集是非频繁项集”这一个性质有效的对频繁项集进行修剪。

首先介绍关联规则分析中的一些基本术语以便对后续详细流程的说明：

1）在关联规则分析中，若一个项集包含 k 个项，则称之为 k-项集。关联规则是形如 $X \rightarrow Y$ 的蕴涵表达式，其中 X 和 Y 是不相交的项集，即 $X \cap Y = \varnothing$。

2）关联规则的强度可以用支持度和置信度两个指标来度量。

对于关联规则 $X \rightarrow Y$，支持度指包含 $X \cup Y$ 的集合个数与总的集合个数的比例，用于确定规则可以用于给定数据集的频繁程度，记作 $s(X \rightarrow Y)$：

置信度指包含 $X \cup Y$ 的集合个数与包含 X 的集合个数的比例，用于确定 Y 在包含 X 的集合中出现的频繁程度，记作 $c(X \rightarrow Y)$：

3）关联规则的发现是指找出支持度大于等于 minsup 并且置信度也大于等于 minconf 的所有规则，其中 minsup 和 minconf 是对应的支持度和置信度的阈值。

本项目使用 matlab 编写基于 Apriori 算法的关联规则分析程序，关联规则分析包含频繁项集的产生和规则的产生两个环节，频繁项集的产生是为了发现满足最小支持度阈值的所有项集，而随后从所发现的频繁项集中提取高置信度的规则作为强关联规则。

（1）产生频繁项集

初始设置最小支持度阈值 minsup，并扫描数据集得到所有的频繁 1-项集的集合 F1，随后开始循环，循环的判断条件为频繁 k-项集 Fk 是否为空集。当频繁 k-项集 Fk 非空时，就使用上一次迭代发现的频繁（k-1）-项集 Fk-1 来产生新的候选项集 Ck，本发明所采用的产生候选项集的方法为合并一对前 k-2 项相同的频繁（k-1）-项集 Fk-1，使用字典序存储项以避免产生重复的候选。在计算候选项集的支持度计数后，删去支持度计数小于 minsup 的所有候选项集，提取当前的频繁 k-项集 Fk。直到没有新的频繁项集产生，即 Fk=Ø 时，输出产生的频繁项集结果，算法结束。

（2）产生规则

本项目选用的算法采用逐层的方式来产生关联规则，其中每层对应规则后件中的项数，其中规则后件即形如 X→Y 的蕴涵表达式中的 Y。初始设置最小置信度阈值 minconf，对每一个频繁 k-项集 Fk（其中 k≥2）均先提取规则后件只含一个项的所有高置信度规则 H1，当 k>m+1 时（其中 m 为规则后件的大小），以 Hm 产生候选项集 Hm+1，并对每个候选项集计算置信度从中提取满足最小置信度阈值 minconf 的规则，循环结束后输出规则，算法结束。

我们对雷雨导致的电压暂降数据进行关联规则分析。设置最小支持度为 0.2，最小置信度为 0.6，得出规则如下：

'暂降幅值 60%-80%=>持续时间 0-0.6，置信度 0.924。这也同样说明了雷电引起的电压暂降事件具有暂降深度低，持续时间短的特点。

6.2 电缆故障导致的暂降事件分析

电缆故障是导致电压暂降的第二大原因。电缆故障主要包括主电缆故障、支线电缆故障、用户电缆故障、电缆中间头故障等。电缆故障导致的电压暂降 ITI 图如下：

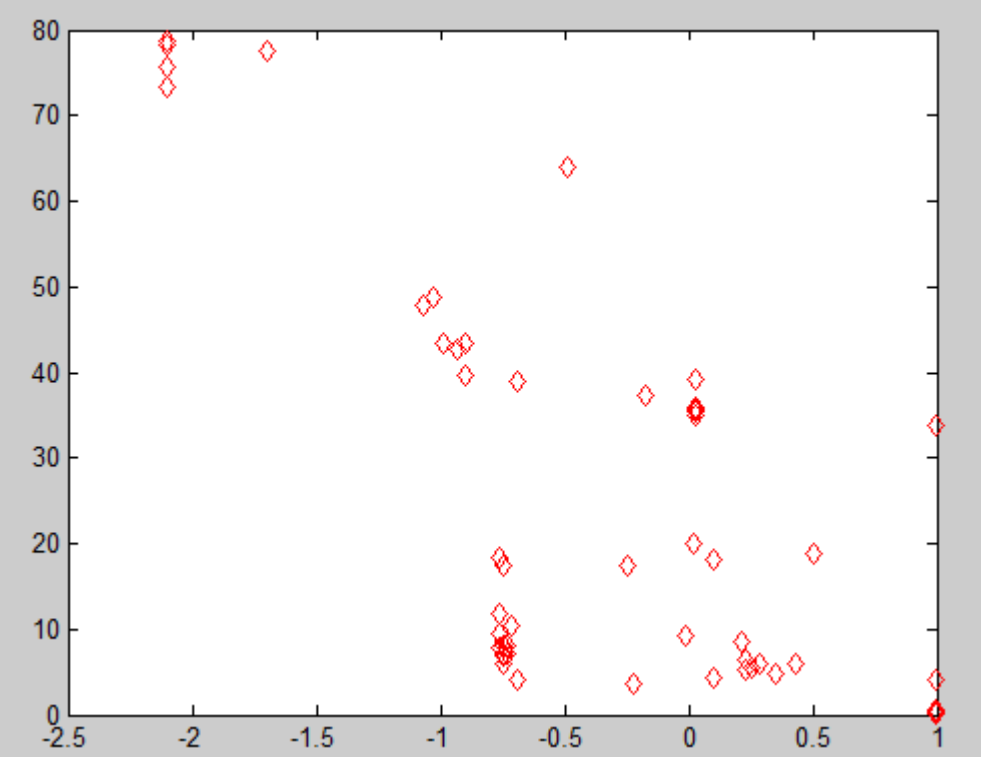


图 6.6 电缆故障导致的电压暂降事件 A 相 ITI 图

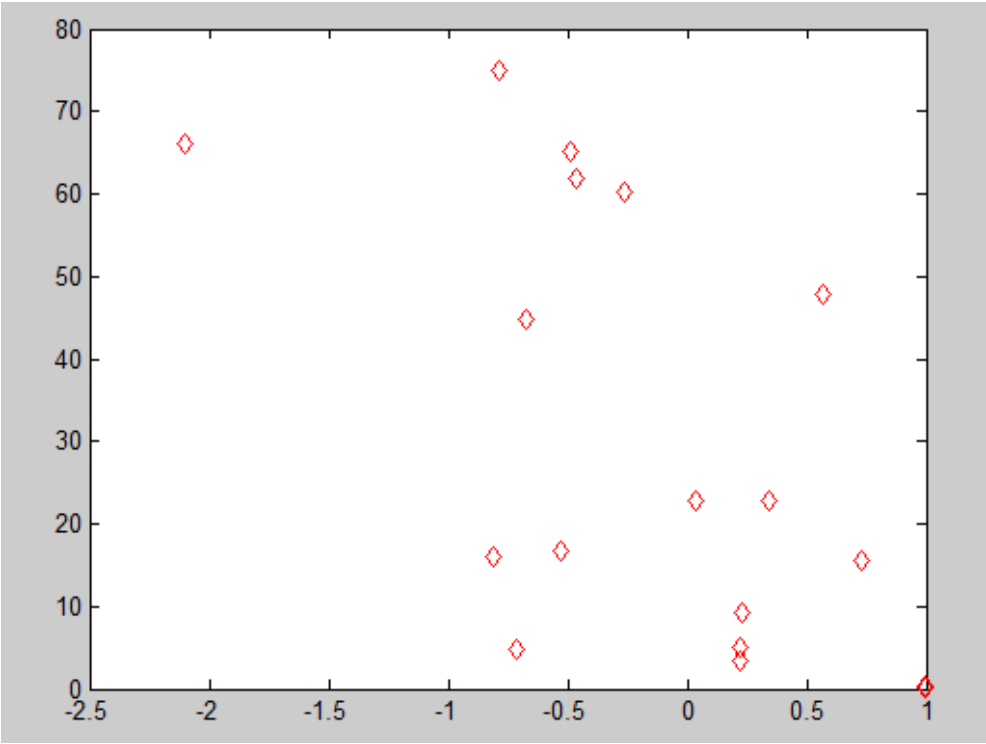


图 6.7 电缆故障导致的电压暂降事件 B 相 ITI 图

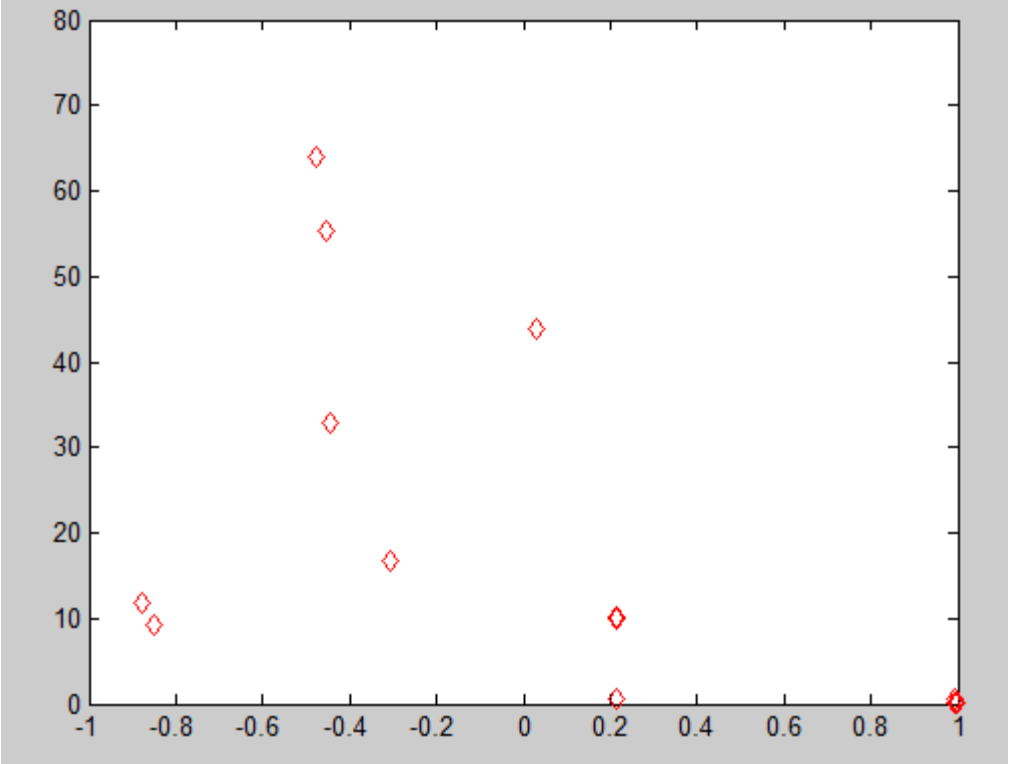


图 6.8 电缆故障导致的电压暂降事件 C 相 ITI 图

通过这三张图我们可以发现，电缆故障导致的电压暂降事件的暂降深度普遍比较高：大多集中在 20 一下；持续时间长（0.1s 以上）。

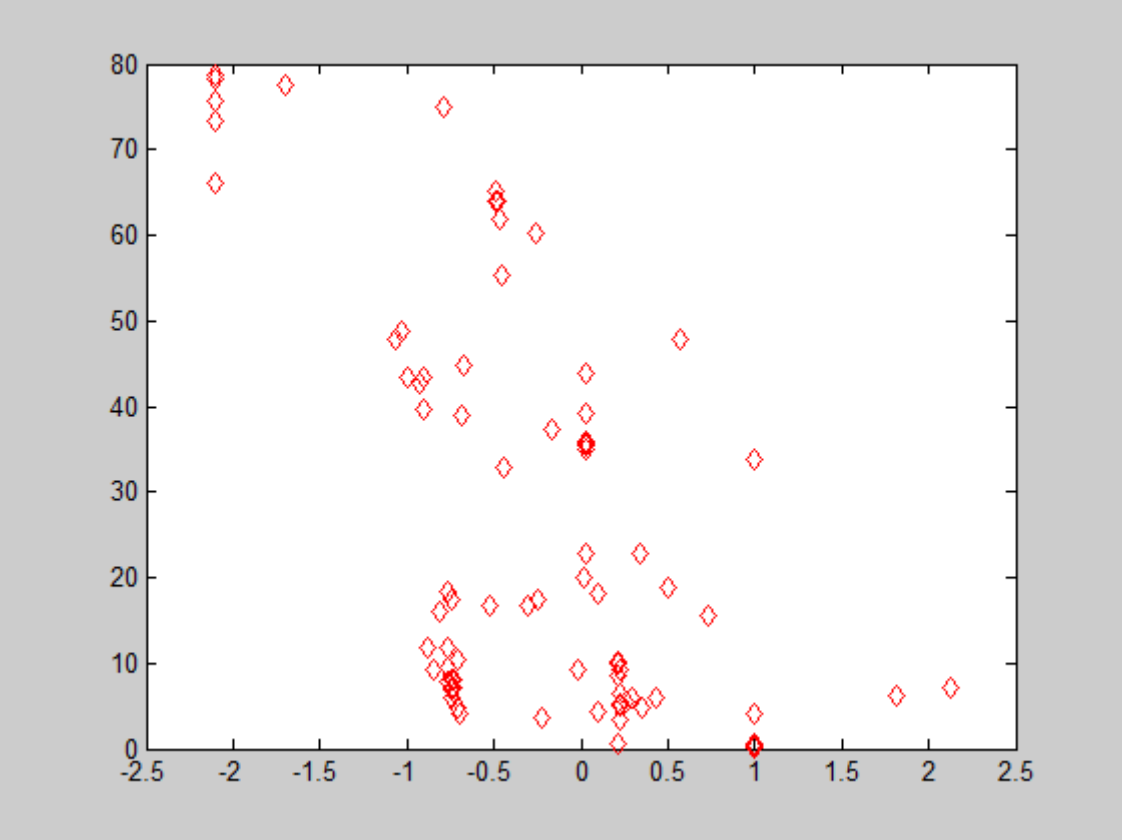


图 6.9 电缆故障导致的电压暂降事件 ABC 三相 ITI 图

6.3 利用支持向量机分类

通过 6.1，6.2 两节的分析可以发现，由雷雨导致的电压暂降事件和电缆故障导致的电压暂降时间具有完全不同的特征：雷雨导致的电压暂降事件深度第，时间短；电缆故障导致的电压暂降时间深度高，时间长。当我们把二者画在一张 ITI 图中，这种对比非常明显：

下图中，蓝色星号代表由雷雨导致的电压暂降事件；红色菱形代表由电缆故障导致的电压暂降事件。横轴是持续时间对数，纵轴是暂降深度。

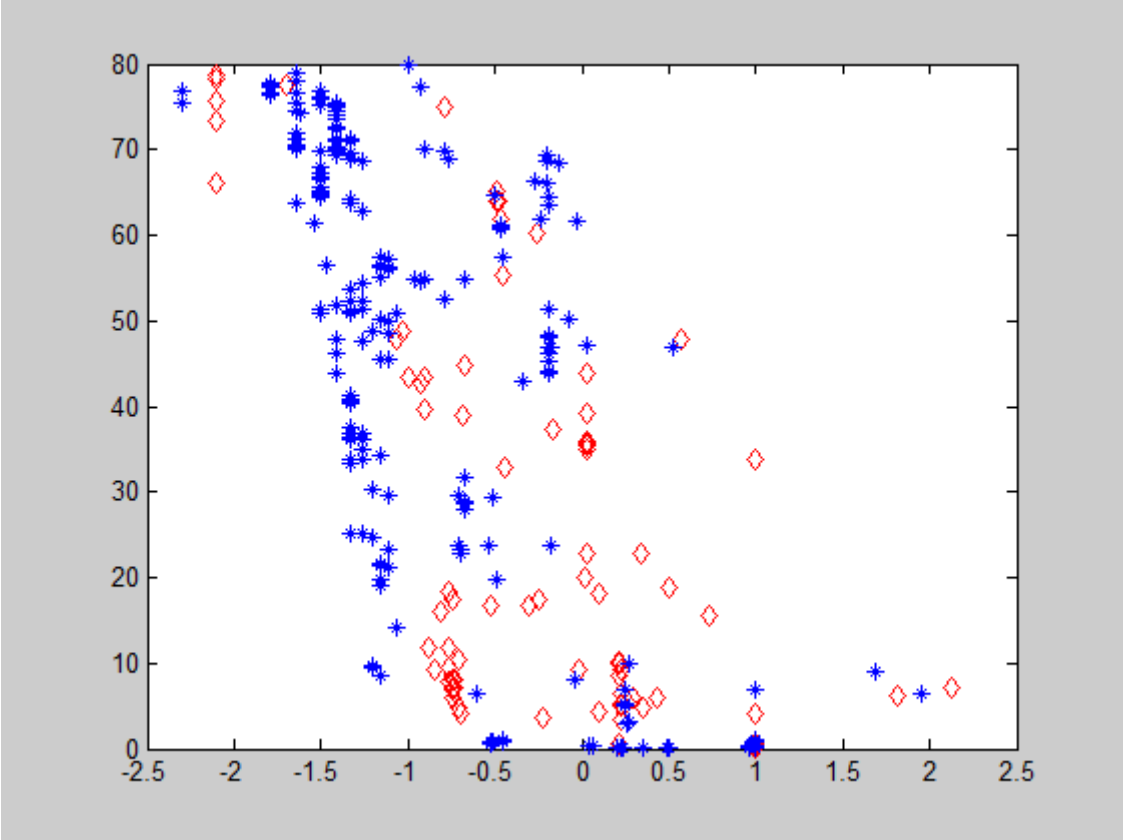


图 6.10 两种故障原因导致的电压暂降对比

如果我们可以根据暂降的特征对暂降的原因进行分类，则可以对未来未知的暂降事件进行分类，对于检修盒电网的维护都有重要的意义。因此，下面利用支持向量机（SVM）方法形成一个超平面，对两种暂态事件进行分类。

SVM 方法是 20 世纪 90 年代初 Vapnik 等人根据统计学习理论提出的一种新的机器学习方法，它以结构风险最小化原则为理论基础，通过适当地选择函数子集及该子集中的判别函数，使学习机器的实际风险达到最小，保证了通过有限训练样本得到的小误差分类器，对独立测试集的测试误差仍然较小。

支持向量机的基本思想是：首先，在线性可分情况下，在原空间寻找两类样本的最优分类超平面。在线性不可分的情况下，加入了松弛变量进行分析，通过使用非线性映射将低维输入空间的样本映射到高维属性空间使其变为线性情况，从而使得在高维属性空间采用线性算法对样本的非线性进行分析成为可能，并在该特征空间中寻找最优分类超平面。其次，它通过使用结构风险最小化原理在属性空间构建最优分类超平面，使得分类器得到全局最优，并在整个样本空间的期望风险以某个概率满足一定上界。

SVM 是从线性可分情况下的最优分类面发展而来的，基本思想可用图 1 来说明。对于一维空间中的点，二维空间中的直线，三维空间中的平面，以及高维空间中的超平面，图中实心点和空心点代表两类样本，H 为它们之间的分类超平面，H1，H2 分别为过各类中离分类面最近的样本且平行于分类面的超平面，它们之间的距离△叫做分类间隔(margin)。

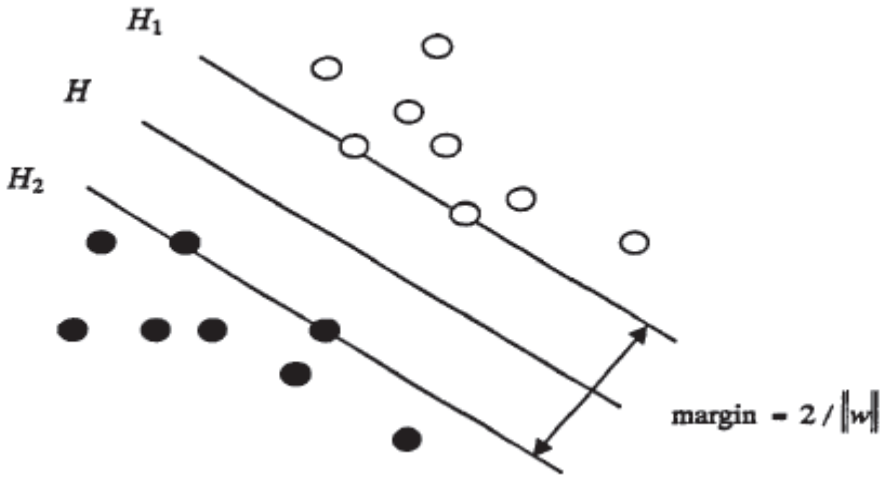


图 6.11 最优分类面示意图

所谓最优分类面要求分类面不但能将两类正确分开，而且使分类间隔最大。将两类正确分开是为了保证训练错误率为 0，也就是经验风险最小(为 O)。使分类空隙最大实际上就是使推广性的界中的置信范围最小，从而使真实风险最小。推广到高维空间，最优分类线就成为最优分类面。

设线性可分样本集为$(\mathrm{x}_i, y_{i_}), i=1,...,n, x\in R^d, y\in\{+1,-1\}$是类别符号。d 维空间中线性判别函数的一般形式为是类别符号。d 维空间中线性判别函数的一般形式为$g(x)=w\cdot x+b$，分类线方程为$w\cdot x+b=0$。将判别函数进行归一化，使两类所有样本都满足$|g(x)|=1$，也就是使离分类面最近的样本的$|g(x)|=1$，此时分类间隔等于$2/\|w\|$，因此使间隔最大等价于使$\|w\|$ (或$\|w\|^2$)最小。要求分类线对所有样本正确分类，就是要求它满足

$$y_i[(w\cdot x)+b]-1\geq 0, i=1,2,...,n$$

满足上述条件(1-1)，并且使$\|w\|^2$最小的分类面就叫做最优分类面，过两类样本中离分类面最近的点且平行于最优分类面的超平面 H1，H2 上的训练样本点就称作支持向量(support vector)，因为它们“支持”了最优分类面。

利用 Lagrange 优化方法可以把上述最优分类面问题转化为如下这种较简单的对偶问题，即：在约束条件，

$$\sum_{i=1}^{n} y_i \alpha_i = 0$$

$$\alpha_i \geq 0, i = 1,2,...,n$$

下面对 α_i 求解下列函数的最大值：

$$Q(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j (x_i x_j)$$

若 α^* 为最优解，则 $w^* = \sum_{i=1}^{n} \alpha^* y \alpha_i$ 即最优分类面的权系数向量是训练样本向量的线性组合。

下图是用支持向量机分类的结果：

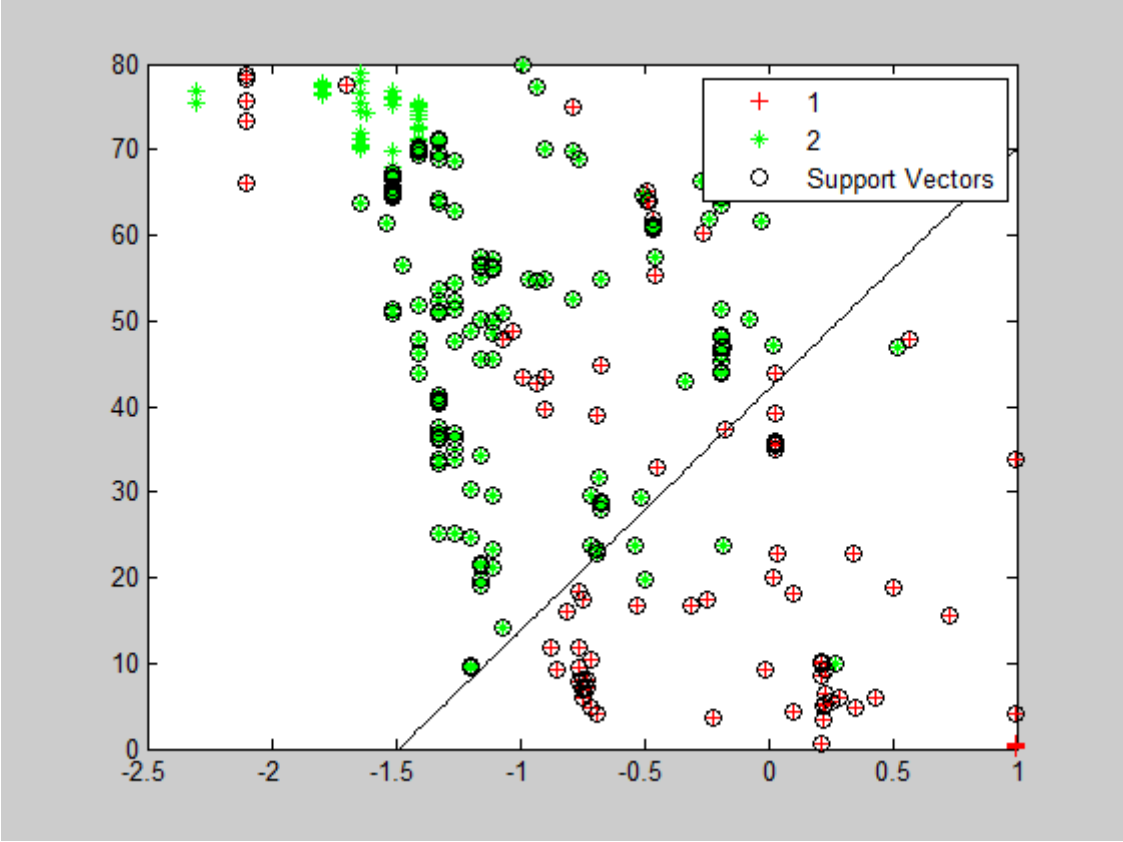


图 6.12 利用 SVM 对两种故障原因导致的电压暂降分类

结　论

本文基于电网对实时性要求高的特点，提出了利用分布式流式计算平台进行电网的大数据分析。流式计算可以大大克服批处理延时长的特点。同时利用开源的数据挖掘平台可以使电力数据挖掘工程师专注于自己熟悉的领域，因为平台可靠性、容错性、可扩展性已经得到了保证。

通过对深圳市电压暂降数据的分析，进行了不同数据源的流聚合、实时电压暂降可视化、电压暂降原因的分类和实时数据压缩研究。通过 SVM 对电压暂降原因的分类，可以有效提取出由雷雨导致的和由电缆故障导致的两类事件的特征。从而为电网的规划、检修等提供意见。由于电网三相之间的高度相关性，基于 PCA 和 PPCA 的数据压缩使得数据存量大大减小，同时，还原后的误差最高只有 0.08%。可以减轻大数据时代电网数据中心的压力和成本。

插图索引

表格索引

参考文献

[1] Dan Vesset, Ashish Nadkarni, etc. Worldwide Big Data Technology and Services 2012–2016 Forecast. IDC Market Analysis, Dec 2012. http://www.idc.com/getdoc.jsp?containerId=238746

[2] Manyika, James, et al. "Big data: The next frontier for innovation, competition, and productivity." (2011).

[3] 国家电网公司大数据应用研究报告，2013.6

[4] 5 Peleg, David. "Distributed computing." SIAM Monographs on discrete mathematics and applications 5 (2000).

[5] Foster, Ian, and Carl Kesselman, eds. The Grid 2: Blueprint for a new computing infrastructure. Elsevier, 2003.

[6] 7 Czajkowski, Karl, et al. "Grid information services for distributed resource sharing." High Performance Distributed Computing, 2001. Proceedings. 10th IEEE International Symposium on. IEEE, 2001.

[7] 8 Armbrust, Michael, et al. "A view of cloud computing." Communications of the ACM 53.4 (2010): 50-58.

[8] 9 Jiawei, Han, and Micheline Kamber. "Data mining: concepts and techniques."San Francisco, CA, itd: Morgan Kaufmann 5 (2001).

[9] Wu, Xindong, et al. "Top 10 algorithms in data mining." Knowledge and Information Systems 14.1 (2008): 1-37.

[10] 11 Quinlan, John Ross. C4. 5: programs for machine learning. Vol. 1. Morgan kaufmann, 1993.

[11] 12 Wagstaff, Kiri, et al. "Constrained k-means clustering with background knowledge." ICML. Vol. 1. 2001.

[12] 13 Suykens, Johan AK, and Joos Vandewalle. "Least squares support vector machine classifiers." Neural processing letters 9.3 (1999): 293-300.

[13] 14 Moon, Todd K. "The expectation-maximization algorithm." Signal processing magazine, IEEE 13.6 (1996): 47-60.

[14] 15 Keller, James M., Michael R. Gray, and James A. Givens. "A fuzzy k-nearest neighbor algorithm." Systems, Man and Cybernetics, IEEE Transactions on 4 (1985): 580-585.

[15] 16 Dugan, Roger C., Mark F. McGranaghan, and H. Wayne Beaty. "Electrical power systems quality." New York, NY: McGraw-Hill,| c1996 1 (1996).

[16] 张伟, and 韩一明. "基于 FPGA 的高速数据采集系统的设计." 电力情报 3 (2002): 46-49.

[17] 18 Chang, Fay, et al. "Bigtable: A distributed storage system for structured data."ACM Transactions on Computer Systems (TOCS) 26.2 (2008): 4.

[18] 19 李翔, and 梁亚丽. "我国电力需求与 GDP 趋势分解比较研究." 现代电力 22.5 (2006): 83-86.

[19] Mori, Hiroyuki. "State-of-the-art overview on data mining in power systems."Power Systems Conference and Exposition, 2006. PSCE'06. 2006 IEEE PES. IEEE, 2006.

[20] Neumeyer, Leonardo, et al. "S4: Distributed stream computing platform." Data Mining Workshops (ICDMW), 2010 IEEE International Conference on. IEEE, 2010.

[21] Conejo, Antonio J., Juan M. Morales, and Luis Baringo. "Real-time demand response model." Smart Grid, IEEE Transactions on 1.3 (2010): 236-242.

[22] Park, Dong C., et al. "Electric load forecasting using an artificial neural network." Power Systems, IEEE Transactions on 6.2 (1991): 442-449.

[23] Hippert, Henrique Steinherz, Carlos Eduardo Pedreira, and Reinaldo Castro Souza. "Neural networks for short-term load forecasting: A review and evaluation." Power Systems, IEEE Transactions on 16.1 (2001): 44-55.

[24] Lee, K. Y., Y. T. Cha, and J. H. Park. "Short-term load forecasting using an artificial neural network." Power Systems, IEEE Transactions on 7.1 (1992): 124-132.

[25] Lu, C. N., H-T. Wu, and S. Vemuri. "Neural network based short term load forecasting." Power Systems, IEEE Transactions on 8.1 (1993): 336-342.

[26] Peng, T. M., N. F. Hubele, and G. G. Karady. "Advancement in the application of neural networks for short-term load forecasting." Power Systems, IEEE Transactions on 7.1 (1992): 250-257.

[27] Negnevitsky, Michael, Paras Mandal, and Anurag K. Srivastava. "Machine learning applications for load, price and wind power prediction in power systems." Intelligent System Applications to Power Systems, 2009. ISAP'09. 15th International Conference on. IEEE, 2009.

[28] Thomassey, Sébastien, and Antonio Fiordaliso. "A hybrid sales forecasting system based on clustering and decision trees." Decision Support Systems42.1 (2006): 408-421.

[29] Mori, Hiroyuki, et al. "Data mining for short-term load forecasting." Power Engineering Society Winter Meeting, 2002. IEEE. Vol. 1. IEEE, 2002.

[30] Tsai, Ronlon, and Jiann-Liang Chen. "Design of a distributed problem-solving system for short-term load forecasting." Circuits and Systems, 1992., Proceedings of the 35th Midwest Symposium on. IEEE, 1992.

[31] Rovnyak, Steven, et al. "Decision trees for real-time transient stability prediction." Power Systems, IEEE Transactions on 9.3 (1994): 1417-1426.

[32] Wehenkel, Louis, et al. "Decision tree based transient stability method a case study." Power Systems, IEEE Transactions on 9.1 (1994): 459-469.

[33] Wehenkel, Louis, and Mania Pavella. "Decision trees and transient stability of electric power systems." Automatica 27.1 (1991): 115-134.

[34] Jack, L. B., and A. K. Nandi. "Fault detection using support vector machines and artificial neural networks, augmented by genetic algorithms." Mechanical systems and signal processing 16.2 (2002): 373-390.

[35] Sheng, Yong, and Steven M. Rovnyak. "Decision tree-based methodology for high impedance fault detection." Power Delivery, IEEE Transactions on 19.2 (2004): 533-536.

[36] Venkatasubramanian, Venkat, R. Vaidyanathan, and Y. Yamamoto. "Process fault detection and diagnosis using neural networks—I. Steady-state processes." Computers & Chemical Engineering 14.7 (1990): 699-712.

[37] Samanta, B., K. R. Al-Balushi, and S. A. Al-Araimi. "Artificial neural networks and support vector machines with genetic algorithm for bearing fault detection."Engineering Applications of Artificial Intelligence 16.7 (2003): 657-665.

[38] Pitt, B. D., and D. S. Kitschen. "Application of data mining techniques to load profiling." Power Industry Computer Applications, 1999. PICA'99. Proceedings of the 21st 1999 IEEE International Conference. IEEE, 1999.

[39] Rodrigues, Fátima, et al. "A comparative analysis of clustering algorithms applied to load profiling." Machine Learning and Data Mining in Pattern Recognition. Springer Berlin Heidelberg, 2003. 73-85.

[40] 3． Mutanen, Antti, et al. "Customer classification and load profiling method for distribution systems." Power Delivery, IEEE Transactions on 26.3 (2011): 1755-1763.

[41] Nizar, Anisah H., Zhao Y. Dong, and J. H. Zhao. "Load profiling and data mining techniques in electricity deregulated market." Power Engineering Society General Meeting, 2006. IEEE. IEEE, 2006.

[42] Catalão, JPSa, et al. "Short-term electricity prices forecasting in a competitive market: a neural network approach." Electric Power Systems Research 77.10 (2007): 1297-1304.

致　　谢

我的本科毕业设计是在曹军威老师的悉心指导下完成的，曹老师从选题、框架到细节，都提出了很多宝贵的意见与建议。他知识渊博、视野广阔，为人为学都是我的榜样。这半年来，曹老师还要求我关注很多学术讲座和国际会议，为我未来研究生阶段的学习打下了很好的基础。

同时，实验室的高田博士、杨明博博士、阳子婧博士和许延祥博士都给了我很多很实用的建议。陈硕师兄非常熟悉 Linux 和开源平台，多次帮我调试系统。在这里我要对他们表示深深地感谢。

转眼间，四年大学时光已匆匆走到尾声。作为清华前百年的最后一届本科生，我在这里度过了一个人一生中学习能力最强，最有活力，最富创造力的阶段。感谢四年来我身边的每一位大牛，虽然经常被虐得“长跪不起”，但也正是这样的环境让我始终以最快的加速度前进，让我水人人的时候都是一种学习。

声　　明

本人郑重声明：所呈交的学位论文，是本人在导师指导下，独立进行研究工作所取得的成果。尽我所知，除文中已经注明引用的内容外，本学位论文的研究成果不包含任何他人享有著作权的内容。对本论文所涉及的研究工作做出贡献的其他个人和集体，均已在文中以明确方式标明。

签　名：_____　　日　期：_____

附录 A 外文资料翻译

S4: 分布式流式计算平台

摘要：

S4 是一个通用的、分布式的、可扩展的、分区容错的、可插拔的流式系统。基于 S4，可以轻松开发处理持续流式数据的应用。在这片论文中，我们详细地描绘了 S4 的框架、多样的应用，包括现实的部署。我们的设计是基于生产环境冲的大规模数据挖掘和机器学习。S4 的设计非常的灵活，可以在商品硬件的大型集群中运行。

关键词：

编程模型；复杂事件处理；并发程序设计；数据处理；分布式编程；map-reduce；中间件；并行编程；实时搜索；软件设计；流式计算

第一章 介绍

S4（简单可扩展流式系统）是一个基于 Mapreduce 模型的分布式流式处理系统。我们设计这个系统来解决现实世界中的问题。应用数据挖掘和机器学习的搜索应用。当今商业的搜索引擎，google，Bing，Yahoo！根据用户的查询提供有组织的网页结果，然后用广告类创造利润。这些广告是基于“每次点击付费”的账单模型。[2]为了把最相关的广告放在最好的位置，科学家创造了一些模型来动态估计每个广告被点击的可能性。可能性的估计是基于用户的偏好、地理位置、最先的查询，最先的点击等。一个典型的搜索引擎每秒可以处理上千个查询。对于每个查询，每页都包含数个广告。为了处理用户的反馈，我们开发了 S4：一个低延时，可扩展，流式处理引擎。

为了使应用在线算法实验更加容易，我们认为一个架构应该同时适用于科研环境和业界环境。科研要求很高的灵活性，对于不同的领域快速开发不同的算

法。这使得利用业务测试在线算法成为可能，最小化费用。对于业界环境，最重要的要求是可扩展（通过增加服务器来增加吞吐量的能力）。我们曾经考虑通过扩展应用开源平台 hadoop 来进行数据流的计算，但我们很快意识到 hadoopp 平台是为批处理设计的。Mapreduce 系统通过调度批处理任务来操作静态数据。在流式计算中，一系列事件以给定速率流入系统中，这个速率我们是不能控制的。处理系统必须跟上这个速率，或者减少负荷。流式处理需要一个和批处理完全不一样的架构。如果尝试搭建一个流式处理和批处理的通用平台，将会非常复杂，而且两方面都不是最优的。Mapreduce 的在线版是 hadoop 的扩展，可以在[3]中阐述

Mapreduce 编程模型可以轻松地使集群中的批处理并行化，并且不同担心系统故障。由于 hadoop 的出现，mapreduce 模型已经从实验室走向了现实世界的应用如网页搜索、骗子检测、网上约会等。尽管如此，在通用的分布式流式计算软件领域，却没有类似的趋势。各种项目和商业引擎如[6][7][8][9][10]，但都被限制在很专业的领域中。Amini[7]提供了各种系统的综述。

诸如实时搜索、高频交易、社交网络等新引用的出现正在突破传统数据处理系统能力的极限。现在需要一个可扩展的流式计算系统，可以处理高数据速率和海量数量。比如，针对个性化的搜索广告，每秒要处理来自报完用户的数以千计的查询。我们发现用户的特征可以帮助增加模型的准确度。

现实世界中的很多系统把流式数据分割成固定的小块，然后利用 Mapredeuce 平台处理。这种方法的缺点是时延和分割块的长度成比例。同时，总开销也与分割动作的开销和初始化的时间成比例。数据块越小，时延越小，总开销越大。最合适的数据块长度取决于应用。因此我们决定开发一个简单的，能实时操作数据流的编程范式，而不是被迫适应 Hadoop 的框架。我们的设计目标如下：

- 提供处理数据流的易用编程接口

- 设计一个具有高可用性和可扩展的集群
- 总过利用每个节点的本地内存最小化时延，避免磁盘的 IO 瓶颈
- 使用一个分散的，对称的体系结构，所有的节点具有相同的功能。没有中心节点。这大大简化了调度和维护的成本。
- 可插拔式的体系结构使得系统尽可能的通用和可定制

为了简化最初的 S4 设计，我们做出如下的假设：

- 允许故障时的数据丢失：一旦一个服务器失败，进程自动移动到备用的服务器。储存在本地内存中的进程的状态，会在传送的过程中丢失。状态会通过输入流重新建立。
- 运行中的集群没有节点的添加和移除

我们发现这些假设对于大部分的应用都是可以接受的。今后，我们将研发不适用这些假设的应用的解决方案。

S4 的和 IBM 流式处理核心中间件（SPC）有很多特性是一样的[7]。两个系统都是为大数据设计，可以从连续的数据流中挖掘出价值与信息。二者最大的不同时架构设计。

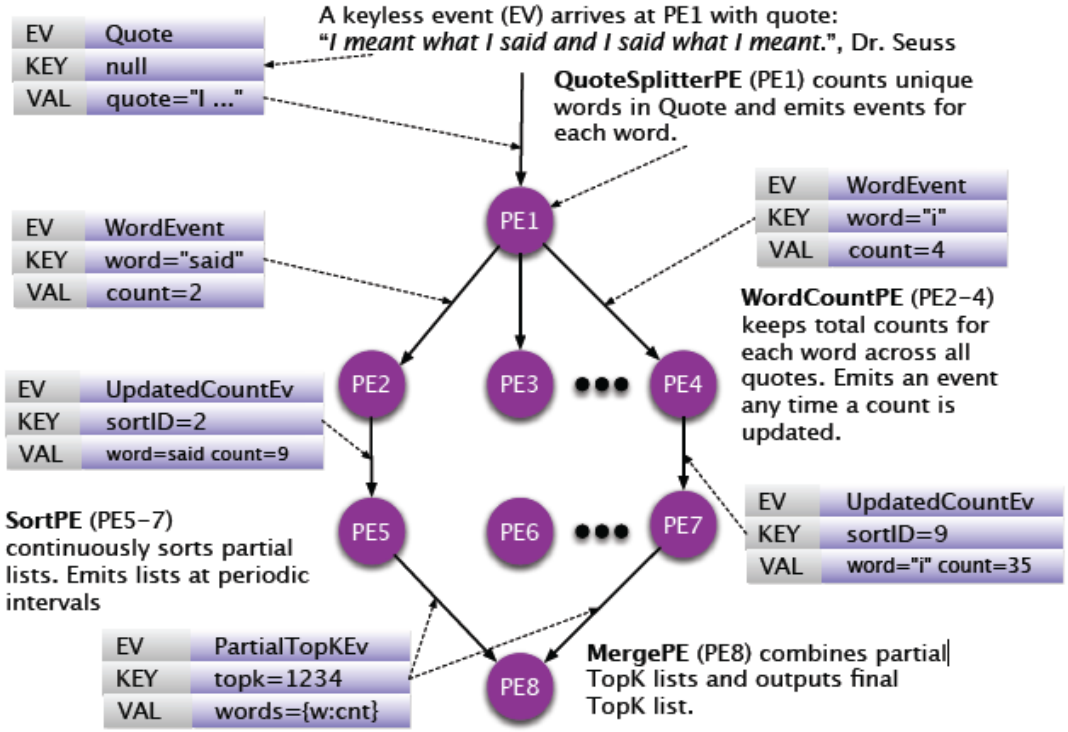
S4 的设计特点有以下几个方面。SPC 主要源自 subscription 模型，S4 来自 Mapreduce 和 Actor 模型的结合。我们相信 S4 更加简化，因为它的对称性。所有的节点都是等同的，没有中心的控制。中心的控制是通过 zookeeper 实现的。Zookeeper 是一个集群管理 服务，应用于很多数据中心的系统中。

第二章 设计

我们定义“流”是一个元素序列。形式是（K，A），K 是键，A 是对应的属性。我们的目标是设计一个灵活的流式计算平台，可以在分布式环境中输入数据流，快速计算出结果，输出数据流。这个部分包含示例应用，然后是 S4 不同组成部分的具体描述。

2.1 例子

图 1 的例子中，输入事件包含了一个英文报价单(Quote)文档。我们的任务是以最小的延迟持续产生一个所有文档范围中出现频率最高的前 K 个单词的排序列表(TOP K)。Quote 事件没有 key，直接发送给 S4。QuoteSplitterPE 对象(PE1)监听 Quote 事件。QuoteSplitterPE 是一个无 key 的 PE 对象，处理所有 Quote 事件。对文档中每一个唯一的 word，QuoteSplitterPE 对象对其计数并发出一个新的 WordEvent 事件，以 word 为 key。WordCountPE 对象监听以 word 为 key 发出的 WordEvent 事件。例如，key 为 word=”said”的 WordCountPE 对象（PE2）接受所有 word=”said”的 WordEvent 类型事件。当一个 key 为 word=”said”的 WordEvent 事件到达，S4 以 word=”said”为 key 查找 WordCountPE 对象。如果 WordCountPE 对象存在，则该 PE 对象被调用，计数增加，否则一个新的 WordCountPE 对像被初始化。每当一个 WordCountPE 对像增加其计数，它就将更新后的计数发给一个 SortPE 对象。SortPE 对象的 key 是一个[1,n]之间的随机整数，n 是想要的 SortPE 对象的总数。当一个 WordCountPE 对象选择了一个 sortID 之后，在其后续生存期就一直使用这个 sortID。使用一个以上 SortPE 对象的目的是为了更好的在多个节点/处理器之间分布负载。例如，key 为 word=”said”的 WordCountPE 对像发送一个 UpdatedCountEvent 事件给一个 key 为 sortID=2（PE5）的 SortPE 对象。每个 SortPE 对象在收到 UpdatedCountEvent 事件时就更新其 TOP K 列表。每个 SortPE 对象定时发送其作为部分的 TOP K 列表给一个单个的 MergePE 对象（PE8），使用任意一个约定的属性键，在这个列子中是 topK=1234。MergePE 对象合并接收到的这些部分列表，然后产出最后的权威 TOP K 列表。



PE ID	PE Name	Key Tuple
PE1	QuoteSplitterPE	null
PE2	WordCountPE	word="said"
PE4	WordCountPE	word="i"
PE5	SortPE	sortID=2
PE7	SortPE	sortID=9
PE8	MergePE	topK=1234

2.2 处理单元

处理单元(PEs)是 S4 中最基本的计算单元。每个 PE 的实例被四个要素唯一标识： (1) 由一个 PE 类和相关配置定义的功能 (2) 它所消费的事件的类型 (3) 这些事件的带 key 的属性 (4) 这些带 key 的属性的属性值。每个 PE 只消费事件类型、属性 key、属性 value 都匹配的事件。它可能会产生输出事件。注意会为每个属性值初始化一个 PE。这个实例化由平台进行。例如，在这个单词计数的例子中，会为输入中的每个单词实例化一个 WordCountPE。每当事件中出现一个新的单词，S4 就为其创建一个新的 PE 实例。

有一种特别的无 key 的 PE，没有属性 key 和属性 value。这种 PE 消费相关类型的所有事件。无 key 的 PE 一般在一个 S4 集群的输入层使用，在这里事件会被赋予一个 key。

有一些内置的 PE 用来处理像 count、aggregate、 join 等等标准任务。许多任务可以使用这些标准 PE 来完成，不需要额外的编码。任务使用一个配置文件来定义。可以使用 S4 的 sdk 很容易的编写定制的 PE。

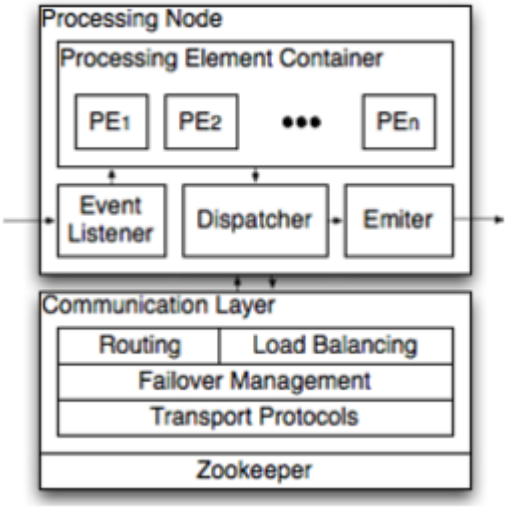
对于有大量唯一 key 的应用，可能有必要随着时间的推移清除 PE 对象。最简单的方法可能是给每个 PE 对象加一个时间戳。如果在一个特定的时间段内没有这个 PE 相关的事件到达，那就可以清除了。当系统内存回收时，PE 对象被清除，其之前的状态也丢失了（在我们的例子中会丢失单词的计数）。这种内存管理策略比较简单，但是并不高效。为了使服务质量(QoS)最大化，我们应该在系统可用内存和对象对系统整体性能影响的基础上最恰当的清除 PE 对象。我们设想一种 PE 对象可以提供其优先级或重要性的方案，这个值是由应用决定的，因此其逻辑应该由应用开发者实现。

2.3 处理节点

处理节点（PNs）是 PE 的逻辑主机。它们负责监听事件，在到达事件上执行操作，通过通讯层的协助分发事件，还有发出输出事件。S4 通过一个哈希函数将每个事件路由到 PN，这个哈希函数作用于事件的所有已知属性值上。单个事件可能被路由到多个 PN 上。所有可能的属性 key 的集合通过 S4 集群的配置获知。PN 中的事件监听器将到来的事件传递给 PE 容器（PEC），PE 容器以适当的顺序调用适当的 PE

有一种特殊的 PE 对象类型：PE 原类型(PE prototype)。它有其身份标识的前 3 个元素（功能、事件类型、属性 key）；而属性值是未赋值的。这个对象在初始化时配置，对于任意值 V，它能够克隆自己来创建相同类型，相同配置，以 V 为属性值的全限定的 PE。PN 对其遇到的每一个唯一属性值触发一次这

个操作。



以上设计的一个结果是，所有包含特定属性值的事件保证会到达特定的相应的 PN，并被路由到 PN 内相应的 PE 上。每个编键的(keyed) PE 能够被映射到一个确定的 PN，映射的规则也是通过一个哈希函数，作用于 PE 的属性值上。无属性键的 PE 会在每个 PN 上被初始化。

2.4 通讯层

通讯层提供集群管理，故障恢复(failover)到备用节点，逻辑节点到物理节点的映射。它自动检测硬件故障并相应的更新这个映射。 发送消息时只指定逻辑节点，发送者不会感知物理节点的存在或故障导致的逻辑节点重映射。 通讯层的 API 提供几种语言的绑定（如 java、C++）。遗留系统可以使用通讯层的 API 以 round-robin 的模式发送输入事件到 S4 集群中的节点。 通讯层使用一个插件式的架构来选择网络协议。事件可能以可靠或不可靠的方式发送。控制消息可能需要可靠发送，而数据消息可能不需要可靠发送以最大化吞吐量。 通讯层使用 ZooKeeper[16]在 S4 集群节点之间进行一致性协作。ZooKeeper 是 Hadoop 下的开源子项目。它为分布式应用提供分布式一致性协作服务。

2.5 配置管理系统

我们预想一个管理系统，在上面操作者可以为 S4 任务创建和销毁集群，并且进行其他的管理操作。分配物理节点到这些 S4 任务集群的操作使用 ZooKeeper[16]进行协作。一个活动节点的集合被分配给特定的任务，剩余的空闲节点仍然留在池中以备需要时使用（例如故障恢复或动态负载均衡）。特别的，一个空闲节点可能同时作为多个不用任务的多组活动节点的冷备。

第三章 编程模型

最上层的编程范例是编写通用的，可重用的，可配置的处理单元，以能够在各种各样的应用中使用。开发者使用 Java 语言编写 PE。PE 通关 Spring 框架被装配到应用中。

处理单元 API 相当简单和富有弹性。开发者本质上只需实现两个主要的操作：一个输入事件操作 processEvent()和一个输出机制 output()。另外开发者可以为 PE 定义一些状态变量。processEvent()在 PE 订阅的每个事件到达时被调用。这个方法实现输入事件处理逻辑，通常为一个内部 PE 状态的更新。output()方法是可选的，可被配置为以各种方式调用。既可以在特定时间间隔 t 调用，也可以在接收到 n 个输入事件时调用。这也意味着它可以每个消息调用一次，即 n==1 的情况。output()方法实现 PE 的输出机制，通常是将 PE 的内部状态发布到一些外部系统。

图 3. QueryCounterPE.java 摘录

```
private queryCount = 0;
public void processEvent(Event event)
{
    queryCount ++;
}
public void output()
{
    String query = (String) this.getKeyValue().get(0);
    persister.set(query, queryCount);
}
```

我们通过一个例子来描述。考虑一个 PE 订阅了用户搜索查询的事件流，从开始计数每个查询实例，并立即将计数值写到一个外部存储。事件流由

QueryEvent 类型的事件组成。类 QueryCounterPE 实现图 3 中描述的 processEvent()和 output()。在这个例子中，queryCount 是 PE 的内部状态变量，持有这个 PE 对查询的计数。最后 PE 的配置在图 4 中描述。在这里，属性 keys 告诉我们 QueryCounterPE 订阅了一个 QueryEvent 类型的事件，并且关注事件的 queryString 属性。配置将 PE 和数据处理组件 externalPersister（这个可以是一个抽象的数据服务系统）绑定起来，并且构造 output()方法为每 10 分钟调用一次。

图 4. QueryCounterPE.xml 摘录

```
<bean id="queryCounterPE"
class="com.company.s4.processor.QueryCounterPE">
   <property name="keys">
      <list>
         <value>QueryEvent queryString</value>
      </list>
   </property>
   <property name="persister" ref="externalPersister">
   <property name="outputFrequencyByTimeBoundary" value="600"/>
</bean>
```

第四章 性能

我们引入了一个现实世界中的基准应用，来描述如何解决 S4 面临的问题，并有一些性能结果。

4.1 点击通过率 (Click-Through Rate) 流计算

用户点击是 Web 上最有价值的用户行为之一。它们提供了用户喜好和当前从事事情的即时反馈，这些信息可以用来在突出的位置显示更受用户欢迎的项目以提高用户体验。在点击付费模式的搜索广告中，发布者，代理商，广告客户基于点击计数决定付费多少。点击通过率(CTR)是点击数除以广告显示数得到的比率。当有了足够的历史数据后，CTR 是用户点击一个项目的可能性的一个很好的估算。精确的来源点击对于个性化和(搜索)排名价值无限，但也受点击欺骗的影响。点击欺骗可以用来操纵一个搜索引擎的排名。点击欺骗一般通过运行在远程电脑(bot)或成群电脑(botnet)上的恶意软件来实现。另一种潜在的威胁

是 impression spam,就是使请求源自 bot。这些请求可能是无恶意的，但是会影响 CTR 估算。

在这个例子中(图 5)，我们演示如何使用 S4 来实时测量 CTR。在搜索广告上下文中，用户查询通过广告引擎处理，返回一个广告排名列表。在例子中，我们为每一个广告查询组合测量 CTR。为了消除点击和显示干扰，我们使用一个启发式规则集合来消除可疑的服务和点击。（在这个例子中，一个服务对应一个用户查询，并被分配一个唯一 ID）。对每个服务，返回一个搜索结果页面给用户，用户可能点击也可能不点击超链接。这个页面相关的点击以唯一 ID 分类）

服务事件包含服务数据，如服务 ID、查询、用户、广告等等，而点击事件只包含点击信息和所点击的服务 ID。在 S4 中计算查询广告级别的 CTR，我们需要使用一个由查询和广告组成的 key 来路由点击事件和服务事件。如果点击负载不包含查询和广告信息，我们需要将上次事件路由到的服务 ID 和查询广告关联作为 key。关联之后，事件必须通过 bot 过滤器。最终，服务和点击聚集到一起来计算 CTR。图 5 显示了一个事件流的快照。

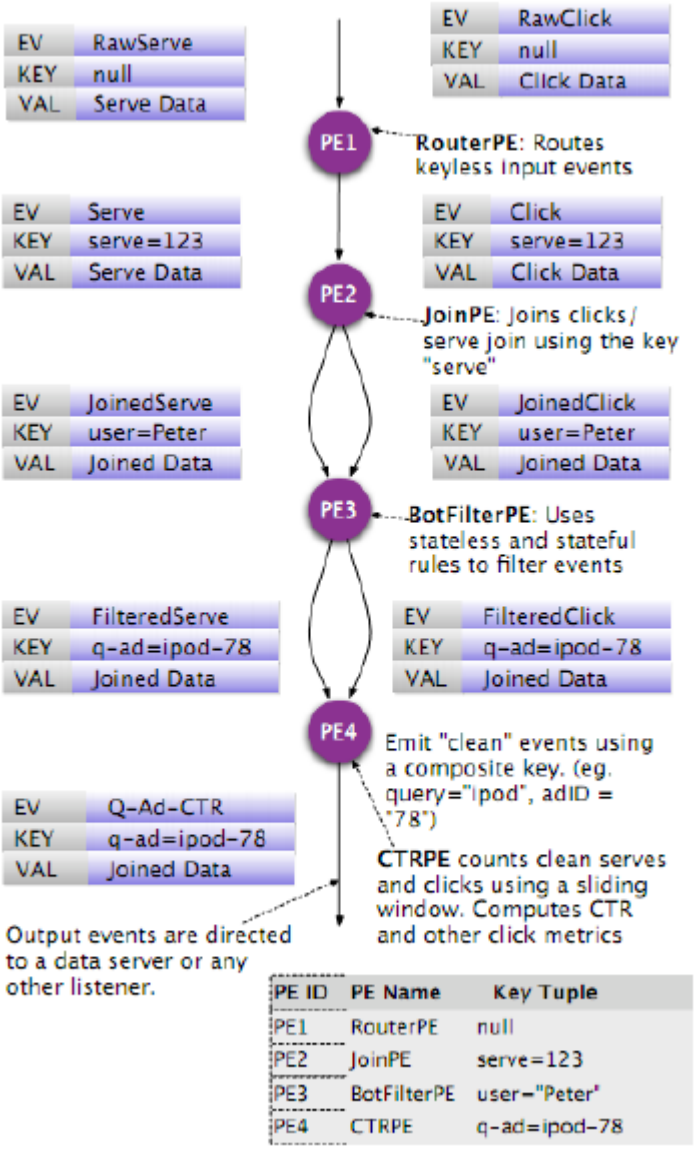


图 5. CTR 计算

4.2 实验搭建

1) 在线实验：我们在实际搜索流量的一个随机抽样上运行 CTR 流计算。为了保证体验的一致性，随机分配到这个实验的搜索引擎用户会再根据他们的浏览器 cookie 的一个哈希做一个修正。平均下来，每天大约有一百万的搜索由 250000 用户发起。实验运行了两个星期，观察到的峰值是每秒 1600 个事件。实验集群有 16 台服务器，每台 4 个 32 位处理器，2GB 内存。

要完成的任务是以非常低的延迟计算查询和广告组合的点击通过率(CTR)。CRT 是一个 24 小时滑动窗口内的累计值。这通过将窗口划分为多个 1 小时的 slot，对每个 slot 的点击和显示做累计来实现。窗口内所有 1 小时 slot 的累计值加起来推送到一个服务系统上。这种方法非常节约内存，但是以更新延迟为代价。使用更多的内存，我们可以维护更小粒度的 slot，例如 5 分钟，从而降低更新延迟。如此实现的系统，提供了短时间段的 CTR 估算，合并后就是长时间段的 CTR 估算。在 PN 故障的时候，我们丢失了那个节点的数据，并且正好路由到那个节点的查询/广告无法做短时间段 CTR 估算。在这种情况下我们的故障策略是放弃(该节点的估算数据)回到长时间段估算

2) 离线实验：我们也运行了一个离线压力测试。我们搭建了 8 个服务器的测试集群，每台服务器 4 个 64 位处理器，16GB 内存。在这些机器上跑了 16 个 PN，每台 2 个。我们使用来自搜索流量 log 的真实点击和服务数据，重建了点击和服务事件来计算搜索查询的真实 CTR，这个会作为精确测试的标准(gold standard)。事件数据由 300 万服务和点击组成。

4.3 结果

在线流量上的实验显示我们可以在不影响收入的前提下将 CRT 提高 3%，主要通过快速检测低质量的广告并把它们过滤出去达成。

离线压力测试的目标是评估系统在远远高于期望的事件流量下的性能。在我们上文描述的测试集群上，我们将离线生成的事件流导向 S2 网格，以一个递增的事件流速率多次运行。在每次运行之后，将系统估算的 CTR 的值和来自搜索 log 的实际 CTR 做比较。表 6 显示了这次测试的结果：

Events per second	Relative Error in CTR	Data Rate
2000	0.0%	2.6 Mbps
3644	0.0%	4.9 Mbps
7268	0.2%	9.7 Mbps
10480	0.4%	14.0 Mbps
12432	0.7%	16.6 Mbps
14900	1.5%	19.9 Mbps
16000	1.7%	21.4 Mbps
20000	4.2%	26.7 Mbps

系统显示了在 10Mbps 左右(性能的)下降。下降的原因归结为 S4 无法足够快地处理这个速率下的事件流，因此导致了事件丢失。

第五章 应用：在线参数调优

这节中，我们引入一个现实中的 S4 应用实例。一个在线参数调优(OPO)系统[17]，使用搜索广告系统的在线流量自动调整一个或多个参数。系统避免了手工调优和持续的人工介入，同时相比于人工操作能在更短的时间内尝试更大范围的参数。系统接受目标系统发出的事件（在我们的列子中是搜索广告系统），测量性能，应用一个适配算法来决定新的参数，然后将新的参数注入回到目标系统。这种闭合循环的方式原理上与传统的控制系统类似。

5.1 功能设计

我们假设目标系统(TS)产生的输出可以用一个流来表示，并且可以通过一个可配置的目标功能(OF)测量其性能。

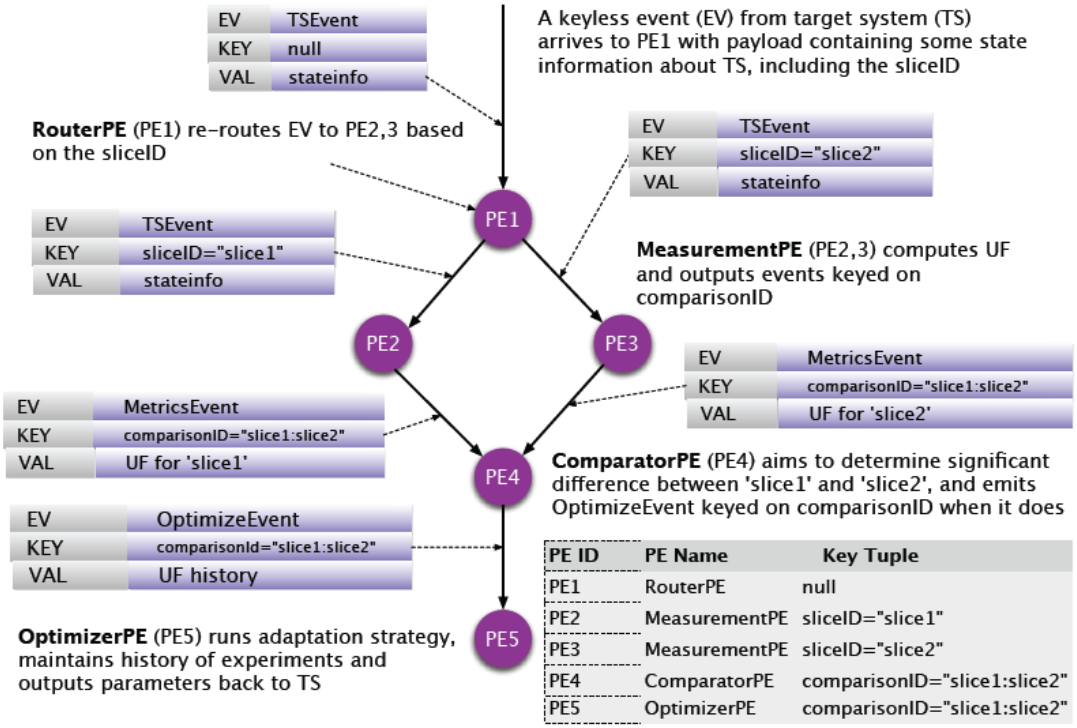
我们将目标系统的输出流随机地分割为 slice1、slice2 两个 slice(分片)。我们要求目标系统能够对每个 slice 应用不用的参数值，并且每个 slice 能够以此标识出自己。

在线参数调优系统(OPO)有 3 个上层功能组件：测量器、比较器和优化器

1）测量器(Measurement)：测量器组件吸收 slice1 和 slice2 流，在每个 slice 中测量 OF 的值。对一个连续的 slot（duration of a slot）测量 OF。slot 可以按时间段来定义，也可以按输出流的事件数定义（连续 n 个事件作为一个 slot）。

2）比较器（Comparator）：比较器组件将测量器的产出作为输入，确定 slice1 和 slice2 的性能是否有统计意义上的不同。当检测到不同时，发送一个消息给优化器。如果在指定个数的 slot 之内没有检测到不同，则声明两个 slice 是相同的。

3）优化器：优化器实现了适配器模式。它将参数影响历史包括最近的比较器输出作为输入，为 slice1 和 slice2 产生新的参数值，这也是新的实验循环开始的信号



5.2 S4 实现

图 7 描述了 OPO 系统在 S4 中的实现。3 个功能组件以 3 个 PE 的形式实现。MeasurementPE 以 sliceID 为 key，slice1 和 slice2 各有一个实例（我们能够很容易的将其扩展为更多 slice 以支持更高级的策略）。目标功能的度量在固定长

度的时间片上进行。ComparatorPE 以 ComparatorID 为 key，对应到一对 slice，在我们的例子中是 slice1，slice2。基于一个对度量结果对的独立的 t-test 来确定两个 slice 之间是否有统计意义上的不同。它被配置为取合格的度量结果中的最小数值。OptimizerPE 以 slice1 和 slice2 为 key。为了实现适配器模式，我们使用一个 Nelder-Mead(aka Amoeba)算法[18]的修改版：一个任意斜率的最小化算法。OTO 系统输出的参数反馈回搜索广告服务系统。这些参数值控制服务系统的外观(aspects)，因此导致不同的用户行为。

5.3 结果

我们在一个搜索广告系统的实际流量分片上运行了 OPO 系统。目标是在当前的参数值（使用传统方法调整）上尽可能地提高系统度量。目标功能是一个搜索引擎上代表总收益和用户体验关系的公式（用户体验和收益的关联性）。流量分片基于搜索引擎用户空间的分布(partitions)：每个 slice 接受大约每天 200000 用户的流量。系统运行了 2 周，尝试调优一个已经知道对搜索引擎性能有重大影响的参数。OPO 系统生成的优化参数值被证明在系统性能的主要度量上有显著的提高：收益提高了 0.25%，点击跳转提高了 1.4%

5.4 总结

我们展示了使用 S4 的一个在线参数调优系统的设计和实现。我们应用这个系统来调整一个搜索广告系统，得到了可喜的结果。这个系统可以用来调整任何有可调参数的动态系统（只要动态系统满足之前描述的几个要求）。

第六章未来的工作

当前系统使用静态路由，通过 ZooKeeper 自动 failover，但是缺乏动态负载均衡和健壮的在线 PE 迁移。我们计划增加这些特性。

参考文献

[1] G. Agha, Actors: A Model of Concurrent Computation in Distributed Systems. Cambridge, MA, SA: MIT Press,1986.

[2] B. Edelman, M. Ostrovsky, and M. Schwarz, “Internet ad-vertising and the generalized second-price auction: Selling billions of dollars worth of keywords,” American Economic Review, vol. 97, no. 1, pp. 242–259, 2007.

[3] T. Condie, N. Conway, P. Alvaro, J. M. Hellerstein,K. Elmeleegy, and R. Sears, “MapReduce online,” EECS Department, University of California, Berkeley, Tech.Rep. UCB/EECS-2009-136, Oct 2009. [Online].

Available: http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-136.html

[4] J. Dean and S. Ghemawat, “MapReduce: simplified data processing on large clusters,” commun. ACM, vol. 51, no. 1,pp. 107–113, 2008.

[5] Apache Hadoop. http://hadoop.apache.org/.

[6] D. C. Luckham, The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2001.

[7] L. Amini, H. Andrade, R. Bhagwan, F. Eskesen, R. King,P. Selo, Y. Park, and C. Venkatramani, “SPC: a distributed,scalable platform for data mining,” in DMSSP ’06: Pro-ceedings of the 4th international workshop on Data mining standards, services and platforms. New York, NY, USA:ACM, 2006, pp. 27–37.

[8] D. J. Abadi, Y. Ahmad, M. Balazinska, U. C ¸etintemel,M. Cherniack, J.-H. Hwang, W. Lindner, A. Maskey,A. Rasin, E. Ryvkina, N. Tatbul, Y. Xing, and S. B. Zdonik,“The Design of the Borealis Stream Processing Engine,” in CIDR, 2005, pp. 277–289.

[9] D. J. Abadi, D. Carney, U. C ¸etintemel, M. Cherniack, C. Con-vey, S. Lee, M. Stonebraker, N. Tatbul, and S. Zdonik,“Aurora: A new model and architecture for data stream management,” The VLDB Journal, vol. 12, no. 2, pp. 120–139, 2003.

[10] Streambase. http://streambase.com/.

[11] M. Stonebraker, U. C ¸etintemel, and S. Zdonik, “The 8 re-quirements of real-time stream processing,” SIGMOD Rec.,vol. 34, no. 4, pp. 42–47, 2005.

[12] S. Schroedl, A. Kesari, and L. Neumeyer, “Personalized ad placement in web search,” in ADKDD ’10: Proceedings of the 4th Annual International Workshop on Data Mining and Audience Intelligence for Online Advertising, 2010.

[13] R. K. Karmani, A. Shali, and G. Agha, “Actor frameworks for the JVM platform: a comparative analysis,” in PPPJ ’09: Proceedings of the 7th International Conference on Principles and Practice of Programming in Java. New York, NY, USA: ACM, 2009, pp. 11–20.

[14] P. Hunt, M. Konar, F. P. Junqueira, and B. Reed,“ZooKeeper: wait-free coordination for internet-scale sys-tems,” in USENIXATC’10: Proceedings of the 2010 USENIX conference on USENIX annual technical conference. Berke-ley, CA, USA: USENIX Association, 2010, pp. 11–11.

[15] K. Gopalakrishna, G. Hu, and P. Seth, “Communication layer using ZooKeeper,” Yahoo! Inc., Tech. Rep., 2009.