摘要

Abstract

# 绪论

本章主要对论文课题的背景进行介绍，对选题进行分析，阐述国内外对聚类算法并行化的研究现状及其在各个领域的应用发展，分析和证明了本课题的研究意义和应用价值，根据以上内容引出本文的结构安排和章节介绍。

## 1.1 课题研究背景及意义

随着计算机硬件的高速发展以及即将到来的AI/VR时代，网络客户端和服务器每时每刻都在产生着浩如烟海的数据，然而我们却缺乏对其充分的理解和应用，传统的数据分析方法已经不能满足海量数据分析和处理的要求。于是，数据挖掘技术应运而生。数据挖掘，比较公认的定义，指的是从大量不完全的、有噪声的、模糊的、随机的数据中，提取出隐含的、事先不为人所知道的、却又是潜在有用的知识和信息的过程。它是一种在海量数据中寻找规则或者模式的过程，是一个新兴的并且具有广阔应用前景的研究领域。

聚类分析是数据挖掘法技术中重要组成部分，可以有效的分析数据并从中发现有用的信息。聚类分析是指根据数据中对象及其之间的关系，将数据对象分组。其目标是，使组内的对象相互之间是相似的（相关的），而不同组中的对象是不同的（不相关的）。组内的相似性（同质性）越大，组间差别越大，聚类效果就越好。它广泛应用于多个领域，如文本聚类、模式识别、人工智能、市场分析、医疗卫生、图像分析和信息检索。

由于数据挖掘是从海量数据中提取有用信息，处理效率问题成了对海量数据处理的瓶颈之一，传统的单机串行算法效率较低；由于部分聚类算法中蕴涵并行性，所以为了解决处理效率问题，将并行化的程序设计思想（并行处理）引入聚类算法，同时降低算法的复杂度，使用集群系统进行并行计算，从而有效的缩短聚类的时间。

Hadoop是一个开源的分布式云计算平台，能够实现对大量的数据集高效、可靠、可伸缩的分布式并行处理。而Hadoop中的MapReduce 编程模式是将已有单机算法实现分布式的关键，通过实现 MapReduce编程模式，我们就可以方便的把已有的算法移植到Hadoop平台实现算法的并行化。

当前，MapReduce在数据挖掘领域被广泛应用，出现了很多基于MapReduce平台的聚类算法。然而随着数据量的进一步增加，实际应用需求的差异，以及实际项目中数据集的不同，针对数据挖掘中的诸多问题，除了研究新的聚类算法以外，针对具体应用需求对现有聚类算法进行改进并移植到 Hadoop 平台上进行分布式实现，从而提高对大规模数据集处理的扩展性，也非常的有效且相对方便，成为当前研究的重要方向，具有十分重大的意义。

## 1.2 国内外发展现状

### 1.2.1 数据挖掘技术的发展

数据挖掘作为一个新兴的多学科交叉应用领域，正在各行各业的决策支持活动中扮演着越来越重要的角色，随着信息技术的迅速发展，各行各业都积累了海量异构的数据资料，这些数据往往隐含着各种各样有用的信息，仅仅依靠数据库的查询检索机制和统计学方法很难获得这些信息，迫切需要将这些数据转化成有用的信息和知识，从而达到为决策服务的目的。数据挖掘分析得到的信息和知识现在已经得到了广泛的应用，例如商务管理、生产控制、市场分析、工程设计和科学探索等。数据挖掘是一个多学科领域，它融合了数据库技术、人工智能、机器学习、统计学、只是工程、信息检索等最新技术的研究成果。

随着海量数据的增加，大数据时代的到来，导致人们对数据的研究和利用越来越多，其中数据挖掘技术的不断进步和发展也给整个世界信息的发展带来了许多成果，在科学领域方面，先进的现代化科学观测仪器的使用造成每天都要产生巨量的数据，如各种同步卫星每小时传回地球的遥感图像数据就达50千兆字节。天文学上有一个很著名的应用系统—SKICAT，这是第一个相当陈宫的数据挖掘应用，也是人工智能技术在天文学和空间科学上第一批成功的应用之一。目前，科学家已利用SKICAT发现了16个新的极其遥远的类星体。在市场营销方面，条形码技术在商业上的普遍使用使得很多行业每天都积累了大量数据，从市场营销来说，通过数据分析了解客户购物行为的一些特征，对提高竞争力及促进销售是有很大帮助的。在金融投资方面，目前国内有很多进行股票分析的软件，并且定期有专家进行股票交易预测。数据挖掘技术还可以应用在甄别诈骗上，进行诈骗甄别主要是通过总结正常行为和诈骗行为之间的关系，得到诈骗行为的一些特征，这样当某项业务符合这些特征时，可以向决策人员提出警告。这方面比较陈宫的系统有FALCON和FAIS系统。在Web应用上，世界上最强大的搜索引擎Google相比其他很多搜索引擎，它的搜索结果更让人满意，其中Google使用的搜索算法主要是PageRank算法，在2001年9月被授予美国专利，Google的PageRank是根据网站的外部链接和内部链接的数量和质量两相衡量网站的价值。

现今，数据挖掘的发展趋势主要在一下几个方面：

数据挖掘语言的标准化：语言的标准化对于数据挖掘系统的开发和数据挖掘技术的普遍使用是至关重要的。其可改进多个数据挖掘系统和功能间的互操作，促进其在企业和社会中的使用。

数据挖掘的可视化：可视化要求已经成为数据挖掘系统中必不可少的技术。可以再发现知识的过程中进行很好的人机交互。数据的可视化起到了推动人们主动进行知识发现的作用。

分布式数据挖掘。分布式技术的到来为日益正常的数据提供了有力的支持，而分布式数据挖掘中将分布式技术和数据挖掘技术的结合，也使对分离数据库的可协作数据挖掘工作开发了一个重要领域。

数据挖掘与数据库系统和Web数据库系统的集成：数据库系统和Web数据库已经成为信息处理系统的主流。数据挖掘系统的理想体系结构是与数据库和数据仓库系统的紧耦合。

挖掘复杂数据类型的新方法：挖掘复杂数据类型是数据挖掘的重要前沿研究课题，也有人称复杂类型的数据挖掘是“下一代数据挖掘”。伴随着数据的增多，需要处理的数据类型也变得越来越发扎，例如数据流、时间序列、时间空间、多媒体和文本数据、虽然现在在很多复杂数据类型的挖掘方面取得了一些进展，但是在应用需求和可用技术之间仍然存在较大的距离。

数据挖掘中的隐私保护和信息安全：随着信息技术的发展，越来越多的数据涌入了网络，其中包括大量电子形式的个人信息，而挖掘技术的发展和科技的更新，在相反的一面上也使大量的个人信息收到了威胁，因此保护隐私的数据挖掘方法愈显重要。

### 1.2.2 MapReduce技术的发展

MapReduce作为云计算的核心技术之一，为并行系统的数据处理提供了一个简单、优雅的解决方案[4]。近几年来，随着数据规模的急速扩大，MapReduce受到了较多的关注，获得了较大的发展，虽然还没有形成成熟的、系统化的理论体系[5]，但机器学习领域中的许多算法都已在MapReduce框架下实现了并行化，并获得了较好的执行效率。

论文[6]针对数据密集型计算环境下数据具有海量、分布、异构、高速变化等特点，分析传统的基于密度的分布式聚类(Density Base Distributed Clustering，DBDC)算法，借助MapReduce编程模型，提出一种新的分布式聚类算法，采用局部和全局的方式处理海量、异构数据，解决具有以上特点的数据密集型计算环境下数据的分析挖掘问题。论文[7]提出了一种基于开源云计算平台Hadoop的网络热点话题发现方案。该方案采用MapReduce分布式并行计算架构处理海量、复杂数据，通过将命名实体词作为文本的特征项，并采用标题和正文的双向量表示文本。实验结果表明随着参与并行计算节点数的增加，话题聚类所用的时间显著下降，因而网络热点话题发现的速度得到明显提高。论文[8]利用MapReduce并行计算模型对Jarvis-Patrick(JP)聚类算法进行并行化设计，用以解决大规模海量文本聚类分析中具有的文本稀疏度和维度高的问题。以搜狗语料库为基础，使用Hadoop平台对算法进行了具体实现，实验结果表明并行运行的Jarvis-Patrick算法与单机环境相比，在处理海量文本数据时拥有更高的运行效率。论文[9]将MapReduce并行计算模型应用于朴素贝叶斯，K-modes和ECLAT算法中。这些算法分别是数据挖掘领域中常用的用以分类，聚类和挖掘频繁项目集的算法。实验结果显示，在确保算法相同准确率的条件下，利用MapReduce模型对算法进行并行化改造可以显著提升算法的执行速度。论文[10]提出一种新的适合于分布式计算的最小生成树算法，结合适合的相似度度量，设计了一种用于解决海量数据分析的分布式聚类算法，并给出了基于MapReduce编程模型的分布式实现。论文[11]提出了一种关系型数据库ChunkDB。该数据库采用了分块结构，是一种分布式数据库，主要用于解决关系型数据库系统与MapReduee并行计算框架之间存在的兼容问题,实验结果表明MapReduce与ChunkDB数据库的结合可以提供高效率的并行化查询。论文[12]设计了一种基于MapReduce计算框架的DeepWeb并行处理算法，将MapReduce并行计算框架应用于DeepWeb爬虫模型。在由虚拟机搭建的计算机集群上对算法进行并行实现，实验结果表明该算法提高了搜索发现数据源的性能，优化了各种资源的使用，证明了将并行处理方法应用于DeepWeb爬虫的有效性。

## 1.3 本文章节安排

本文将分为六个章节对课题进行阐述，各章的内容安排如下：

第一章：绪论。本部分作为开篇伊始，将着重带领大家了解本课题选用的价值及其背景和意义。通过介绍与课题相关的国内外最新发展现状，包括数据挖掘学科的进展，MapReduce技术的进步与发展，以及包括本节在内的整篇文章的章节介绍，以便读者更容易的把握本文主要内容。

第二章：相关技术研究。本章节将介绍本课题的研究路线，及其中用到的关键技术，对此进行详细的阐述。分别介绍数据挖掘的分类与研究，对聚类分析模块做进一步的深入探索，对目前常用的聚类方法进行对比分析，然后解释了MapReduce的编程模型，对其中的主要过程进行了介绍，最后介绍了供热锅炉系统的物理模型，方便读者了解供热锅炉系统的运作方式。

第三章：基于曲线相似度的供热过程评价。本章提出了一个基于曲线相似度的供热过程评价模型，通过Frechet距离，伸缩和平移一共五个度量对出水温度和评价曲线做对比，最后加权平均为一个度量，量化的评价供热锅炉的供热过程，从而方便锅炉管控人员根据数据操作锅炉输出参数，进行指标结算，更加精确的控制热量，节省能耗。

第四章：基于MapReduce的并行K均值算法研究。本章在之前的基础上，对聚类算法进行并行化处理，研究了基于划分的常规聚类算法K均值，又称k-means，对其进行并行处理从而提高传统串行算法的效率，解决在处理大规模数据集时的时间复杂度问题。

第五章：实验及结果。本章内容在前几章对研究内容进行分析完毕后的算法和过程进行实现，通过搭建Hadoop平台，运行于Linux环境中，利用集群的高并发效果编写相应的MapReduce函数，同时对天津地区供热锅炉系统的历史数据进行分析，进行聚类同时对离群点进行各个维度的回溯，从而发现各个锅炉房管理人员的管控问题，帮助管理者进行绩效管理。

第六章：供热效果实时监测系统。考虑到供热系统的即时性特征，为了方便管理着可以及时发现问题，同时可以利用数据可视化的手段将聚类的结果进行展示，方便人们查看具体的内容理解数值型数据的直观含义，帮助管控人员及时发现问题，调整控制策略。

第七章：总结和展望。最后将本篇文章所涉及的研究路线和方法进行总结，通过指出目前算法和系统中的不足之处，期待后续在各个方面进行仔细的调整。

# 相关技术研究

## 2.1 数据挖掘

数据挖掘是从海量数据中挖掘出有趣的模式和知识的过程。其中数据源可能会包括数据仓库、数据库、Web存储、内存存储或动态的进入系统的数据。这里存在大量的数据挖掘模式，包括特征化与区分，频繁模式、关联和相关性挖掘，分类与回归，聚类分析，离群点分析。一般而言，这些任务可以分为两类：描述性和预测性。描述性挖掘任务刻画目标数据中数据的一般性质。预测性挖掘任务在当前数据上进行归纳，以便做出预测。

类/概念性描述：特征化与区分。数据特征化是目标类数据的一般特性或特征的汇总。通常通过查询来收集对应于用户指定类的数据。将数据汇总和特征化有一些有效的方法。基于统计度量和图的简单数据汇总，基于数据立方体的OLAP上卷操作可以用来执行用户控制的、沿着指定维的数据汇总。面向属性的归纳技术可以用爱进行数据的泛化和特征化，而不必一步步地与用户交互。数据特征化的输出可以用多种形式提供，例如饼图、条图、曲线、多维数据立方体和包括交叉表在内的多维表。数据区分是将目标类数据对象的一般特性与一个或多个对比类对象的一般特性进行比较。目标类和对比类可以由用户指定，而对应的数据对象可以通过数据库查询检索。

挖掘频繁模式、关联和相关性。频繁模式是在数据中频繁出现的模式。存在多种类型的频繁模式，包括频繁项藉、频繁子序列和频繁子结构。频繁项藉一般是指频繁的在事务数据集中一起出现的商品的集合，子结构可能涉及不同的结构形式，可以与项集或子序列结合在一起，如果一个子结构频繁的出现，则称它为结构模式。挖掘频繁模式导致发现数据中有趣的关联和相关性。通常，一个关联规则被认为是无趣的而被丢弃，如果它不能同时满足最小支持度阈值和最小置信度阈值。还可以做进一步分析，发现相关联的属性-值对之间的有趣的统计相关性。

用于预测分析的分类与回归。分类是这样的过程，它找出描述和区分数据类或概念的模型，以便能够使用模型预测类标号未知的对象的类标号。导出模型是基于对训练数据集的分析。该模型用来预测类标号未知的对象的类标号。分类预测类别标号，而回归建立连续值函数模型。也就是说，回归用来预测缺失的或难以获得的数值数据值，而不是（离散的）类标号。术语预测可以指数值预测和类标号预测。尽管还存在其他方法，但是回归分析是一种最常使用的数值预测的统计学方法。

聚类分析。不像分类和回归分析标记类的数据集，聚类分析数据对象，而不考虑类标号。在许多情况下，开始并不存在标记类的数据。可以使用聚类产生数据组群的类标号。对象根据最大化类内相似性、最小化类间相似性的原则进行聚类或分组。也就是说，对象的簇这样形成，使得相比之下在同一个簇中的对象具有很高的相似性，而与其他簇中的对象很不相似。所形成的每个簇都可以看做一个对象类，由它可以导出规则。聚类也便于分类法形成，即将观测组织成类分层结构，把类似的事件组织在一起。

离群点分析。数据集中可能包含一些数据对象，他们与数据的一般行为或模型不一致。这些数据对象是离群点。大部分数据挖掘方法都将离群点视为噪声或异常而丢弃。然而，在一些应用中，罕见的事件可能比正常出现的事件更令人感兴趣。离群点数据分析称作离群点分析或异常检测。

## 2.2 聚类分析

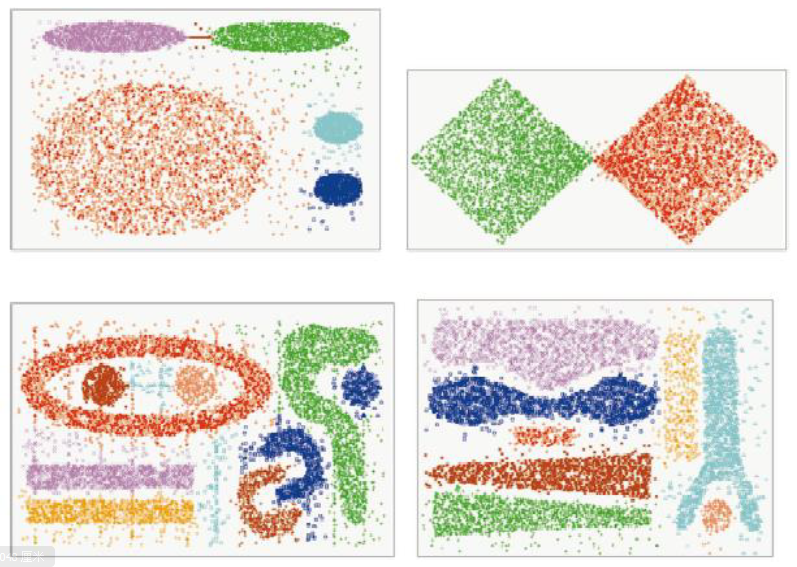
### 2.2.1 聚类分析的概念

聚类分析简称聚类，是一个把数据对象（或观测）划分成子集的过程。每个自己是一个簇，使得簇中的对象彼此相似，但与其他簇中的对象不相似。由聚类分析产生的簇的集合称作一个聚类。在这种语境下，在相同的数据集上，不同的聚类方法可能产生不同的聚类。划分不是通过人，而是通过聚类算法进行。聚类是有用的，因为它可能导致数据内事先未知的群组的发现。

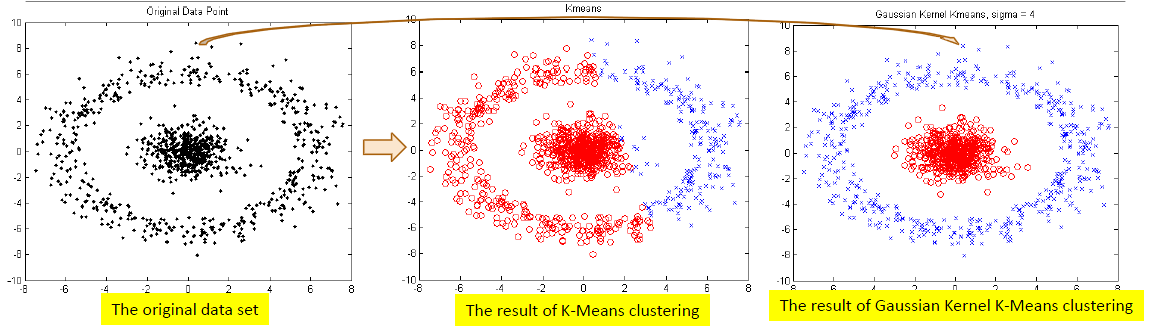
作为一种数据挖掘功能，聚类分析也可以作为一种独立的工具，用来洞察数据的分部，观察每个簇的特征，将进一步分析集中在特定的簇集合上。另外，聚类分析可以作为其他算法的预处理步骤，之后这些算法将在检测到的簇和选择的属性或特征上进行操作。

在某些应用中，聚类又称作数据分割，因为它根据数据的相似性把大型数据集合划分成组。聚类还可以用于离群点检测，其中离群点（“远离”任何簇的值）可能比普通情况更值得注意。

### 2.2.2 基于层次的聚类算法

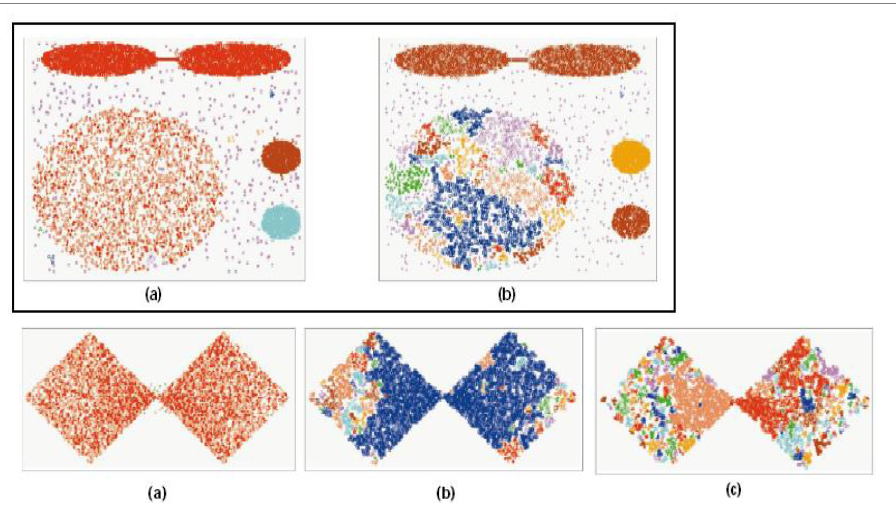
层次聚类方法将数据对象组成层次结构或簇的“树”。该方法主要有两种路径：凝聚和分裂，也可以理解为自底向上法（bottom-up）和自顶向下法（top-down）。凝聚法就是一开始每个个体（object）都是一个类，然后根据相似性度量寻找同类，最后形成一个“类”。分裂法就是反过来，一开始所有个体都属于一个“类”，然后根据相似性度量排除异己，最后每个个体都成为一个“类”。这两种路径本质上没有孰优孰劣之分，只是在实际应用的时候要根据数据特点以及自己想得到的“类”的个数，来考虑是自上而下更快还是自下而上更快。至于根据相似性度量判断“类”的方法就是最短距离法、最长距离法、中间距离法、类平均法等等（其中类平均法往往被认为是最常用也最好用的方法，一方面因为其良好的单调性，另一方面因为其空间扩张/浓缩的程度适中）。基于层次的方法中比较新的算法有BIRCH（Balanced Iterative Reducing and Clustering Using Hierarchies）主要是在数据体量很大的时候使用，而且数据类型是数字型；ROCK（A Hierarchical Clustering Algorithm for Categorical Attributes）主要用在categorical的数据类型上；Chameleon（A Hierarchical Clustering Algorithm Using Dynamic Modeling）里用到的相似性度量是kNN（k-nearest-neighbor）算法，并以此构建一个graph，Chameleon的聚类效果被认为非常强大，比BIRCH好用，但运算复杂度很高，O(n^2)。以下是Chameleon的聚类效果图，其中一个颜色代表一类，可以看出来是可以处理非常复杂的形状的。

### 2.2.3 基于划分的聚类算法

其原理简单来说就是，想象你有一堆散点需要聚类，想要的聚类效果就是“类内的点都足够近，类间的点都足够远”。首先你要确定这堆散点最后聚成几类，然后挑选几个点作为初始中心点，再然后依据预先定好的启发式算法（heuristic algorithms）给数据点做迭代重置（iterative relocation），直到最后到达“类内的点都足够近，类间的点都足够远”的目标效果。也正是根据所谓的“启发式算法”，形成了k-means算法及其变体包括k-medoids、k-modes、k-medians、kernel k-means等算法。k-means对初始值的设置很敏感，所以有了k-means++、intelligent k-means、genetic k-means；k-means对噪声和离群值非常敏感，所以有了k-medoids和k-medians；k-means只用于数值类型数据，不适用于类别类型数据，所以k-modes；k-means不能解决非凸（non-convex）数据，所以有了kernel k-means。另外，很多教程都告诉我们基于划分的聚类多适用于中等体量的数据集，但我们也不知道“中等”到底有多“中”，所以不妨理解成，数据集越大，越有可能陷入局部最小。下图显示的就是面对非凸，k-means和kernel k-means的不同效果。

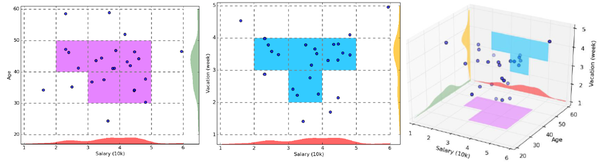
### 2.2.4 基于密度的聚类算法

由于k-means解决不了这种不规则形状的聚类。于是就有了基于密度的聚类算法来系统解决这个问题。该方法同时也对噪声数据的处理比较好。其原理简单说画圈儿，其中要定义两个参数，一个是圈儿的最大半径，一个是一个圈儿里最少应容纳几个点。最后在一个圈里的，就是一个类。DBSCAN（Density-Based Spatial Clustering of Applications with Noise）就是其中的典型，可惜参数设置也是个问题，对这两个参数的设置非常敏感。DBSCAN的扩展叫OPTICS（Ordering Points To Identify Clustering Structure）通过优先对高密度（high density）进行搜索，然后根据高密度的特点设置参数，改善了DBSCAN的不足。下图是表现了DBSCAN对参数设置的敏感。



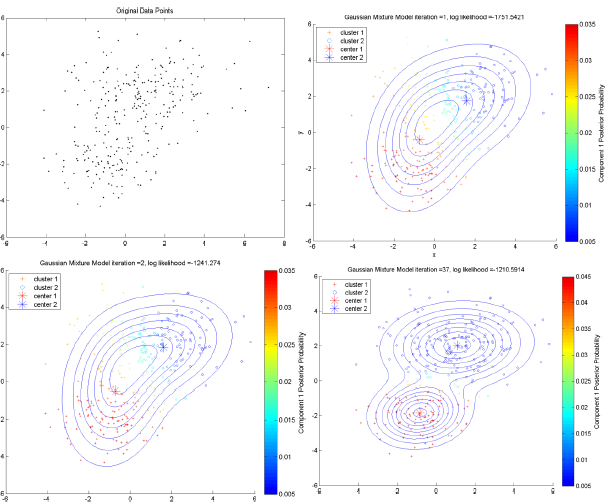
### 2.2.5 基于网格的聚类算法

基于网格的聚类算法原理就是将数据空间划分为网格单元，将数据对象集映射到网格单元中，并计算每个单元的密度。根据预设的阈值判断每个网格单元是否为高密度单元，由邻近的稠密单元组形成“类”。该类方法的优点就是执行效率高，因为其速度与数据对象的个数无关，而只依赖于数据空间中每个维上单元的个数。但缺点也是不少，比如对参数敏感、无法处理不规则分布的数据、维数灾难等。STING（STatistical INformation Grid）和CLIQUE（Clustering In QUEst）是该类方法中的代表性算法。下图是CLIQUE的一个例子：



### 2.2.6 基于模型的聚类算法

基于模型的聚类算法主要是指基于概率模型的方法和基于神经网络模型的方法，尤其以基于概率模型的方法居多。这里的概率模型主要指概率生成模型（generative Model），同一“类”的数据属于同一种概率分布。这中方法的优点就是对“类”的划分不那么“坚硬”，而是以概率形式表现，每一类的特征也可以用参数来表达；但缺点就是执行效率不高，特别是分布数量很多并且数据量很少的时候。其中最典型、也最常用的方法就是高斯混合模型（GMM，Gaussian Mixture Models）。基于神经网络模型的方法主要就是指SOM（Self Organized Maps）了，也是我们所知的唯一一个非监督学习的神经网络了。下图表现的就是GMM的一个例子，其中用到EM算法来做最大似然估计。



## 2.3 MapReduce

MapReduce是一个编程模型，也是一个处理和生成超大数据集的算法模型的相关实现。用户首先创建一个Map函数处理一个基于key/value pair的数据集合，输出中间的基于key/value pair的数据集合；然后再创建一个Reduce函数用来合并所有的具有相同中间key值得中间value值。现实世界中有很多满足上述处理模型的例子，也就是说MapReduce模型的提出，是基于现实世界中能满足这种处理模型的实际情况的存储。

其原理是：利用一个输入key/value pair集合来产生一个输出的key/value pair集合。MapReduce库的用户用两个函数表达这个计算：Map和Reduce。用户自定义的Map函数接受一个输入的key/value pair值，然后产生一个中间key/value pair值得集合。MapReduce库把所有具有相同中间key值I得中间value值结合在一起后传递给reduce函数。

用户自定义的Map函数接受一个中间key的值I和相关的一个value值得集合。Reduce函数合并这些value值，形成一个较小的value值得集合。一般的，每次Reduce函数调用只产生0或1个输出value值。通常我们通过一个迭代器把中间value值提供给Reduce函数，这样我们就可以处理无法全部放入内存中的大量的value值的集合。

# 基于曲线相似度的供热行为评价方法

随着供热行业自动化水平的不断提高，“煤改气”和“十三五”规划节能减排的进行，我国的城市供热过程基本上实现了自动控制，供热品质得到改善，能源利用率得到提高。很多学者已经利用数据挖掘等技术对锅炉系统历史数据分析。孙群丽等对锅炉运行数据进行关联规则挖掘，提供了几组在不同负荷及外部条件下的最优运行方式与参数控制[1]；路海昌等通过对时间序列进行相空间重构，建立了基于支持向量回归的时序数据预测模型，从而实现对锅炉输出参数的预测[2]；岳晓忠采用后向反馈BP神经网络理论和关联规则算法的数据挖掘方法，对锅炉实时运行数据进行分析，从而建立锅炉运行模型[3]。以上的分析都是如何去优化控制锅炉，但实际操作人员由于各种原因较难掌握这些方法的使用，而供热公司也没有合适的方法对现场人员供热过程量化分析，即缺乏切实可靠的量化评价方法，难以满足管理者实时掌握考评状况和调整运行策略的需求[4]。因此研究供热过程的评价方法对推进节能减排、降低运行成本都具有重要意义。

本章首先解释目标温度曲线的由来并分析目标温度曲线与实际出水温度曲线之间不同的多种可能情况，然后将其分解为趋势变化、平移和伸缩三个属性相似度并给出各个属性的相应计算方法，并最后融合为一个一致性度量来评价供热过程。通过实验分析，证明了此评价方法的有效性，为相关管理人员量化管理提供了一种参考依据，避免仅以燃气、水、电等能耗来衡量供热行为，而是需要区分不同的供热过程，在满足供热户室内温度的情况下尽可能节能。

## 3.1 目标温度曲线与出水温度曲线分析

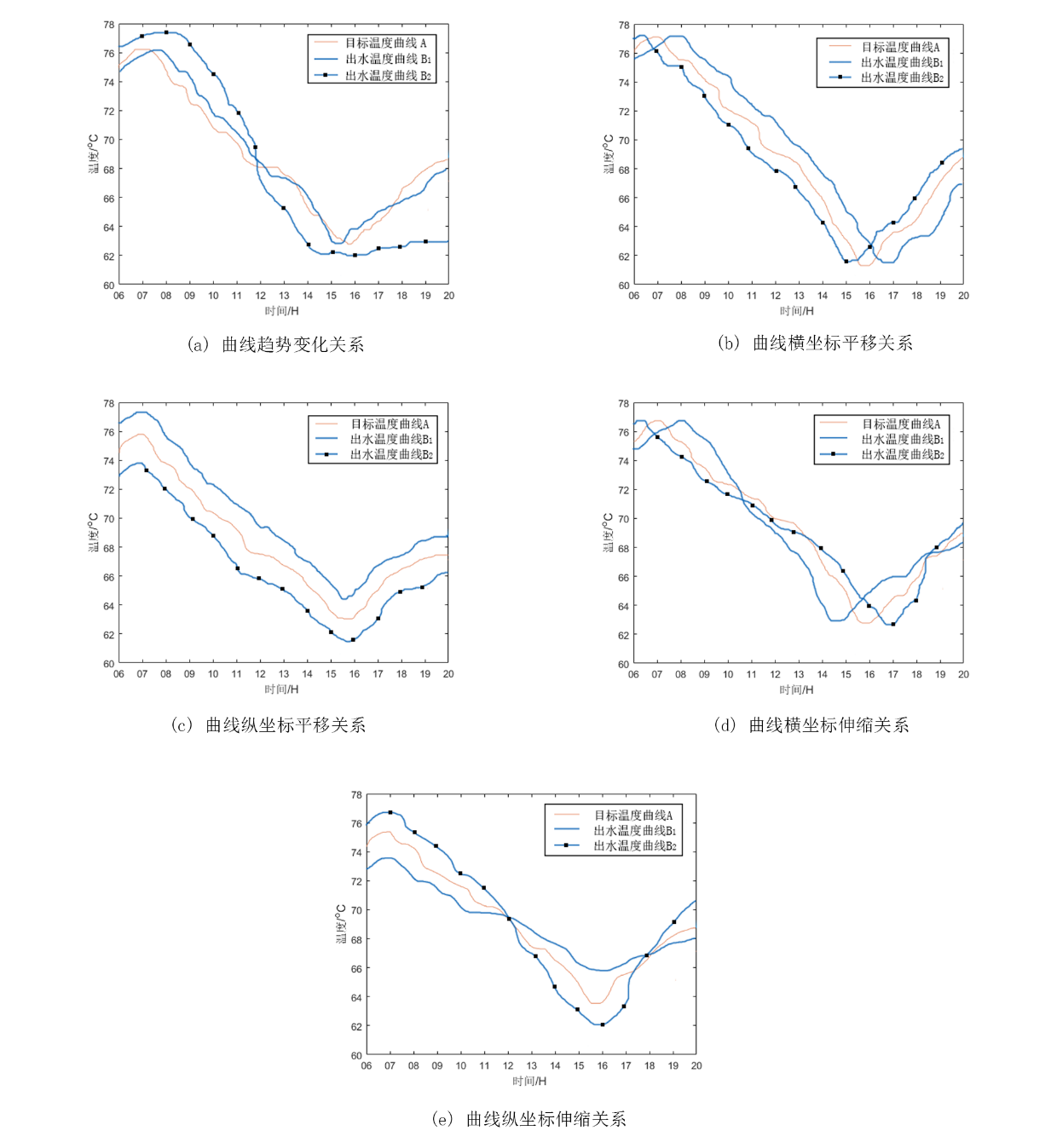
### 3.1.1 目标温度曲线

早期锅炉系统管理人员根据运行经验，根据室外温度会制定一个出水温度标准作为供热锅炉出水温度的参考，通常与室外温度为线性相关。在此基础上一段连续时间的出水温度就构成了目标温度曲线。随着节能减排的规划以及物联网技术的应用，目标温度的定义需考虑各方面因素确定，主要包括天气条件（室外温度、风速、日照），供热用户不同时间段的需求，回水温度，出水提前量等。其中天气条件将直接影响供热用户的采暖需求，回水温度反映了热量的利用情况，而供热公司也需要根据供热用户的作息规律、生活习惯、上班或在家的情况进行适度的调控，尽量节约能源，出水提前量指热水热交换后到用户家里的时间。因此是一个比较复杂的过程，需要考虑较多因素，主要因素如公式（1）所示。

式中为时间，分别为时间的室外温度、风速、日照、偏移量、回水温度、出水提前时间。

目标温度曲线主要是根据专家经验综合以上因素并参考供热用户建筑计算或对大量历史采集数据分析预测得到。

### 3.1.2 目标温度曲线与出水温度曲线相似度



将目标温度曲线（A）与出水温度曲线（B）进行一致性分析，可以分析出锅炉房供热过程是否严格按照要求供热，及满足节能要求，为锅炉管理人员对操作人员量化管理提供参考。

两条曲线的一致性存在多种情况，如图1所示。其中图1-(a)中曲线B1基本与曲线A相同，而曲线B2的在12时后较曲线A的趋势有明显差异，表明曲线B2基本没有按照曲线A进行相应的调整，即两条曲线反映的供热调节趋势不一致；图1-(b)表现了曲线间的横向平移问题，在供热数据上，横坐标的平移代表了出水温度与目标温度调整的提前和延时情况，纵坐标的平移代表了出水温度与目标温度的温差情况，即供热温度相比目标温度偏高或偏低。其中曲线B1在横坐标上较曲线A整体提前，而曲线B2在横坐标上较曲线A整体延后，表明虽然趋势调整基本正确，但调节的及时性存在问题；图1-(c)表现了曲线间的纵向平移问题，曲线B1在纵坐标上较曲线A低了近2摄氏度，表明供热温度偏低，未能达到供热户供暖需求，而曲线B2在纵坐标上较曲线A高了近2摄氏度，表明供热温度偏高，造成了能源浪费；图1-(d)中，曲线B1的最大值与最小值间横坐标差距相较于曲线A较小，说明该降温时间延后，而升温时间提前，造成了能源的浪费，而曲线B2的最大值与最小值间横坐标差距大于曲线A，说明降温时间提前而升温时间延后，未能达到供热户需求，容易引发客诉；图1-(e)中，曲线B1与曲线A相比，其最大值与最小值间的纵坐标差距更小，在高温时未能达到供热户需求，在低温时浪费热量，而曲线B2最大值与最小值间的纵坐标差距更大，在高温时浪费热量，在低温时未能达到供热户需求。

综合上述分析，两条曲线间主要存在趋势变化、平移和伸缩三个属性上不同。因此，出水温度曲线与目标温度曲线的一致性评价问题可以分解为对这三种属性相似度的计算问题。在供热数据中，曲线间的差异也代表了不同的供热行为：趋势变化主要反映两者在整体形态上的一致性，即操作人员能否按照要求进行合理的调控，横向时间的偏移体现了温度调整的提前或延后，纵向温度的偏移体现了供热温度的偏高或偏低；伸缩体现了曲线在升温和降温调整的不同。

## 3.2 供热效果评价

### 3.2.1 出水温度曲线与目标温度曲线的趋势变化

两条曲线趋势变化相同，可以认为锅炉系统能综合考虑各种因素对出水温度做出合理的调整，使供热户在不同的条件下得到最合适的热量。对两条曲线趋势的分析，可以认为是曲线的相似性问题[5]。

曲线的相似性测度一般有两种方法：距离测度法和相似性函数法[6]。相似性函数是用函数的方法来表征两曲线相似的程度，主要有夹角余弦和相关系数等方法，但由于供热数据受天气和地域影响较大，很难拟合成相应的曲线函数，而距离测度法主要有Euclidean距离、Minkowsky距离、Hausdorff距离、Fréchet距离等[7]，其中，Hausdorff距离和Fréchet距离主要用来计算两个点集间的相似性，但Hausdorff距离忽略了点集的时间序列问题，基于供热系统时序数据的特点，本文采用Fréchet距离作为曲线趋势变化属性相似度的度量，其优点在于充分考虑了曲线的连续性，非常适用于曲线间的相似性比较[8]。

Fréchet 距离由 M. Fréchet提出，描述了两质点分别沿着2条给定曲线以任意速度单向运动时，二者之间的最短距离。Axel Mosig和Michael Clausen曾将Fréchet 距离与变换群的交叉子集结合，应用到判别两条曲线的相似性上[9]，曹凯等引入Fréchet距离进行云规则推理，设计了一种智能地图匹配算法[10]。Eiter 和 Mannila在连续 Fréchet 距离的基础上提出了离散 Fréchet 距离[11]的定义 , 而朱洁等考虑了离散Fréchet距离的关键特征峰值点，减少了算法的复杂度并将其运用到了手写签名验证上[12]，收到了一定的效果。

离散Fréchet距离定义如下：

（1） 给定1个有n个至高点的多边形链，1个沿着P的k步，分割P的峰值点成为k个不相交的非空子集，使得和

（2） 给定2个多边形链，，1个沿着A和B的组合步是1个沿着A的k步和1个沿着B的k步组成，使得对于，中有1个恰好包含1个至高点。

（3） 1个沿着链A和B的组合步的花费(cost)为：

其中dist()为a,b间的欧氏距离，则链A和B间的离散Fréchet距离为：

### 3.2.2 出水温度曲线与目标温度曲线的平移问题

出水温度曲线与目标温度曲线的平移表现为两者横纵坐标的差异，在温度曲线中，横坐标的度量为时间，表现为出水温度调整的提前或延后问题；而纵坐标的度量为温度，体现在两条曲线的温差问题，具体为锅炉系统是否能按需达到预定温度以及在不需要较多热量时降低负荷节约能源。

#### 3.2.2.1时间差异

时间差异定义为两条曲线的n个同一维度上特征点间的时间差均值。而特征点的确定会对度量结果产生较大影响，考虑到供热锅炉数据的特点，在同一维度上很难找到成对的特征点[13]。这里先获得两条曲线的最值，然后按照最值将每条曲线单独划分为多个单调区间，对于单调增区间最小值为初始特征点对，否则最大值为初始特征点对，然后在各区间根据初始特征点对的类型计算下一个特征点对，取其时间的差值作为时间差异。依次计算出每个时间段的差异集合，最后将这些差异的均值作为差异度量。

式中为特征点的个数，m为每个单调区间的点数，A为目标温度，B为出水温度，则为目标温度曲线在第j个区间温度为第i大值的时间，供热控制较差时可在合理时间段取值。

#### 3.2.2.2 温度差异

温度差异简称温差，主要表现在供热温度较低时是否能满足采暖需求，反之是否发生能源浪费。这里将分两方面考虑，曲线的最大值差和最小值差。

两条曲线最大值处的温差，能够判断供热效果，出水温度是否能按需达到采暖需求，为供热用户提供足够的热量。目标温度较高时通常意味着用户在家或者气象条件不能提供较多自然热量，所以需要锅炉系统高负荷运转提供充足热量，也可以一定程度上减少客诉。

两条曲线最小值处的温差，能够判断供热锅炉系统是否节约能源。目标温度较低时通常意味着，用户家中无人，或者室外温度等气象条件能提供较多自然热量，故锅炉系统需要降低运行负荷减少热量，节约能源降低运行成本。

综合上述两个方面将两条曲线的温差定义如下：

A为目标温度，B为出水温度，为目标温度曲线最小值点的温度。

### 3.2.3 出水温度曲线与目标温度曲线的伸缩问题

出水温度曲线与目标温度曲线的伸缩问题，在其横坐标上表现为锅炉系统在时间上，是否按统一节奏对锅炉系统进行调控，对天气情况的变化是否做出时间一致的操作；在其纵坐标上表现为锅炉系统对温度控制的灵敏度，在高低温转换时可以及时达到预期温度。

借助离差标准化的思想，最值差可以完整的表现整体的数据跨度[14]，将横纵坐标的最值差比作为两条曲线的伸缩比，能较好的反映数据整体的特点，对其横向和纵向伸缩比的计算方式如式（5）和（6）所示.

式中为目标温度曲线最大值的横坐标， 为出水温度曲线最小值的纵坐标。

### 3.2.4 评价结果

将三种属性相似度共五个度量加权融合为出水温度曲线和目标温度曲线的一致性度量：

式中、、、、分别为趋势、横向平移、纵向平移、横向伸缩、纵向伸缩属性相似度的权值，，可通过数据统计及最小二乘法得出[15]。 、 、 、 、分别为五个属性相似度的阈值。

本文通过分析出水温度曲线与目标温度曲线间的一致性，分别给出曲线的趋势变化、平移、伸缩三种属性相似度的计算方式，并将其加权融合为一个评价结果，用来对锅炉供热过程进行评价。

# 基于MapReduce的并行K均值算法研究

## 4.1 k均值算法研究

K均值（K-means）由MacQueen最早提出，它是一个基于划分的聚类算法的直接实现，介于其算法思想简单，串行算法的实现相对容易的特点，在多个计算机交叉学科里面都有广泛的应用。它是聚类算法中最常见的划分方法，所谓的划分方法就是给定一个包含 n个数据对象的数据集，将数据集划分为k 个子集，其中每个子集均代表一个聚类，同一聚类中的对象相似度较高，而不同聚类中的对象相似度较小。聚类相似度是利用各聚类中对象的均值获得一个“中心对象”来进行计算。划分方法也就是说将数据分为 k组，这些组满足以下要求：

1. 每组至少应该包含一个对象；
2. 一般的划分中每个对象必须且只能属于一个组；
3. 在一些模糊划分中可以允许每个对象属于多个组。

k 均值算法的工作过程如下：首先从 n 个数据对象中任意选择 k 个对象作为初始的聚类中心，而对于所剩下的其他对象，则根据它们与这些聚类中心的相似度，一般相似度是根据具体需要确定，比如对于文本的聚类多采用余弦相似度，而对于数据的聚类可以采用欧氏距离计算相似度。计算相似度之后分别将他们分配给与其最相似的聚类；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相似度函数开始收敛为止。

|  |
| --- |
| 算法：根据聚类中的均值进行聚类划分的 k均值算法 |
| 输入：聚类个数 k，以及包含 n 个数据对象的数据集  输出：满足方差最小标准的 k个聚类  处理流程：   1. 从 n 个数据对象中任意选择 k 个对象作为初始聚类中心。 2. 分别计算剩下的数据对象到这 k 个中心的距离或者相似度，根据实际需求来选择距离度量。 3. 将这些数据对象分别划归到距离最近或者相似度最高的聚类。 4. 根据聚类结果，重新计算 k 个聚类各自的中心，计算方法是取簇中所有元素各自维度的算术平均数。 5. 将数据集中全部数据对象按照新的中心重新聚类。 6. 重复以上操作，经过 t 次迭代计算直到聚类结果不再变化或者变化趋于指定的收敛域为止。 7. 7、聚类结束，将结果输出。 |

## 4.2 k均值算法MapReduce并行化

根据算法思想设计算法的流程，主要设计Map函数、Combine函数以及Reduce函数分别实现途中的算法执行过程。

### 4.2.1 Map函数的设计

Map函数的任务是先构建一组全局的初始聚类中心，Map函数将文件的每行作为一个样本，从样本中提取出有用的数据，以key/value键值对形式表示，并计算每个样本到各个聚类中心的距离，选择距离最小的聚类中心，并把该数据样本分配到该聚类中，并把该数据样本标记为所属的新聚类类别，形成key/value键值对的输出形式。Map过程的输入为待聚类所有数据对象和上一轮迭代后产生的聚类中心，<key, value>对是MapReduce框架默认的形式，输入数据以<行号, 记录行>的形式表示<key’, value’>对；Map函数根据全局的聚类中心计算得到与每个输入样本距离最近的聚类中心，并做新聚类的标记；输出中间结果以<聚类ID, 记录属性向量>的形式表示<key, value>对。Map函数输入的是当前记录相对于输入数据文件起始点的偏移量，value是当前记录的各维坐标值。先从value中解析出当前记录各维的值；然后分别计算其与k个聚类中心的距离，找出距离最近的聚类ID;最后输出<key’, value’>，其中key’是距离最近的聚簇ID，value’是当前记录的各维坐标值。

Map函数的输入数据存储在每一台主机的硬盘中，其文件格式为一个接一个的<key, value>对，每一个<key, value>对即为一个输入数据，另外，每台主机还拥有一个全局的聚类中心表。根据这两个信息，函数可以计算得到与每个数据对象最为相似的聚类中心。Map函数的输出数据同样为<key’, value’>对的形式，key’表示聚类ID; value’表示与该聚类中心最为相似的数据对象。Map函数的伪代码如表所示：

|  |
| --- |
| 算法：Map函数 |
| map(<key, value>,<key’, value’>)  {  定义instance数组，记录从value中解析出每个样本的各个为维度值  minDis = Math.max(); 辅助变量minDis初始化为最大值  index = -1; index初始化为-1；  for(i = 0; i <= k – 1; i ++) do {  dis = instance 与第i个聚类中心各维度值得距离；  if dis < minDis{  minDis = dis;  index=i;  }  }  将index作为key’;  将各维坐标值作为value’;  输出<key’, value’>;  } |

为了方便中间结果在本地进行预处理，以此减轻集群的时间耗费和通信压力，一般会在Map操作之后插入一个Combine操作，但Combine操作也不是每个程序都适用的。将函数输出数据<key’, value’>对进行本地合并，相当于本地化的操作。由于Map的输出数据都暂存在本地的HDFS分布式文件系统中，所以添加Combine操作可以减小集群的通信量和传输所耗费的时间，同时加快下一步Reduce操作的执行时间。

### 4.2.2 Combine函数的设计

Combine函数的作用就是对Map过程产生大量中间结果进行本地化处理，可以减轻数据在节点之间的传输时间耗费和带宽占用。Combine函数对所处节点内的Map结果进行预处理，对具有同一key值的value值进行处理，然后将处理得到的局部聚类结果，紧接着，再传给集群中的Reduce函数进行规约操作，这就是Combine函数的任务[63]。Combine函数的输入形式为key/value键值对，key代表聚类类别ID，value代表聚类为key的记录的各维坐标值。Combine函数首先解析出每个记录的各维坐标值，然后，将各维坐标值相加，得到局部聚类结果的累加和，同时计算得到总的样本个数。输出的<key’, value’>对中key’是聚类ID;value’是记录总数和各维坐标值的累加和。

Map任务完成后，MapReduce算法会启动一个Combine任务来合并那些具有相同聚类ID的中间结果数据。由于中间结果数据都是本地化存储的，该过程不会产生集群的通信开销。在Combine函数中，对本地的具有相同聚类ID的记录向量进行求和。为了得到新的全局聚类中心，还需要统计节点中每个聚类的记录个数。Combine函数的伪代码如表：

|  |
| --- |
| 算法： Combine函数 |
| combine(<key, value>,<key’, value’>)  {  初始化一个用于存储各维坐标值累加的数组，初始值都为0；  初始化变量num为0，用于统计相同聚类的记录数目；  while(value.hasNext()){  解析value中每个记录的各维坐标值；  将各维坐标值累加存放到数组中；  num++;  }  将key作为key’;  构造存储num和数组信息的字符串，作为value’;  输出<key’, value’>;  } |

### 4.2.3 Reduce函数的设计

Reduce函数通过汇总Combine函数得到的局部聚类结果计算出新的聚类中心，并将其用于下一轮迭代运算[64]。Reduce函数输入数据的形式为key/value键值对，key代表聚类类别ID，value代表各个Combine函数得到的中间结果(intermediate result)，Reduce函数首先计算每个节点输出的局部聚类结果的样本个数，并解析每个样本的各维坐标值，然后将对应的各维累加值分别对应相加，再除以刚才计算得到的总样本个数，计算结果就是新的聚类中心坐标。输出结果的形式为key/value键值对，key代表聚类类别ID，value代表计算得到的新聚类中心。

Reduce过程的输入是Combine过程之后得到的局部聚类结果，将得到的局部聚类结果进行合并，生成全局聚类结果。Reduce函数的输入是从Combine任务得到的每个节点的数据，该数据包括了同一类中的记录向量的累加和和以及记录个数。然后，将所有节点上同一聚类的记录向量累加并计算类中总的记录个数，得到新的聚类中心，作为下一轮迭代运算的聚类中心。Reduce函数的伪代码如表所示：

|  |
| --- |
| 算法： Reduce函数 |
| reduce(key, value),<key’, value’>  {  初始化一个用于存储各维坐标值累加的数组，初始值都为0；  初始化NUM为0，用于统计相同聚类的总的记录个数；  while(value.hasNext()) {  解析value中的各维坐标值和记录个数num;  将各维坐标值累加存放到数组中；  NUM+=num；  }  将数组中的各个分量除以NUM，得到新的聚类中心；  将key作为key’;  构造一个包含新聚类中心各维坐标值的字符串，作为value’;  输出<key’, value’>;  } |

在执行完Reduce任务之后，根据Reduce的输出结果计算新的聚类中心，并更新到HDFS分布式文件系统中，并将该文件复制到集群中的所有节点上，然后，计算连续两轮MapReduce Job的误差平方和准则函数，若差值小于设定的值，则聚类准则函数已收敛，算法结束；否则，将新的酒类中心替换原来的聚类中心，启动新一轮迭代运算，同样是Map任务，Combine任务和Reduce任务的流程。当迭代输出趋于稳定收敛时，就可以得到最终的聚类结果。

### 4.2.4 算法复杂度分析

在给出详细的分析之前，先假设计算数据对象与聚类中心距离的时间复杂度是O(1)，每个数据对象的维数d，各维要分别计算与k各聚类中心的距离这个过程的时间复杂度为O(nkd)，其中n为对象个数。每次迭代过程中，每个数据对象被归到离它最近的聚类中心所在类，然后根据新的聚类结果计算新的聚类中心，这个过程的时间复杂度为O(nd)，因此，一次迭代总的时间复杂度可认为O(nkd)，因此，总的时间复杂度为O(nkdt)，其中t为迭代次数；所以，时间复杂度通常可用于O(n)表示。

K-means聚类MapReduce并行化的计算过程中，假设m是集群中节点的数目，计算是在m个节点上均衡的执行，在每次迭代中，每个数据对象聚类计算的时间复杂度是O(kd)，所以每次迭代中所有数据对象距离计算的时间复杂度是O(nkd)。在理想情况下，所有数据对象的距离计算平均的分布到m个节点上并行执行，所以时间复杂度被缩小为O(nkdt/m)。因此，随着集群中节点数目的增加，每个节点聚类计算所消耗的时间也会大大减少。

# 实验及结果

## 5.1 Hadoop集群环境的搭建

### 5.1.1 Hadoop集群资源规划

为了充分利用有限的实验资源，将使用四台虚拟机搭建在VMware环境中，在宿主机中将硬盘资源和内存切分为多个虚拟机进行实验，其宿主机和虚拟机的配置分别如下：

宿主机：硬盘 700GB 内存 16GB CPU I7 4760四核处理器 操作系统Windows 10 企业版

虚拟机：硬盘 20GB 内存 1GB CPU I7 4760单核处理器 操作系统 Linux Kali

根据以上配置内容，规划的拓扑结构如下图所示：

### C:\Users\NSSA\AppData\Local\Temp\enhtmlclip\13134833-69706f5f43ef4920861cab670e300b47.jpg5.1.2 Hadoop环境配置

1. 使用对Windows支持较好的VMware软件将宿主机分割，依据IP地址分配虚拟机地址

2. 为每台虚拟机安装Linux系统，这里选用Kali，使用的版本是kali-linux-1.0.9a-i386，Kali的前身是backtrack，是一个基于Debian的Linux操作系统，内置诸多易用的黑客工具，开放超级管理员权限，并裁剪内核

3. 由于实验所用宿主机为本机，为了便捷访问各个节点，需要安装SecureCRT连接工具，方便进行上传文件和访问节点操作

4. 使用CRT等软件登录虚拟机，安装常用的vim、ssh等软件

sudo apt-get install vim

sudo apt-get install ssh

5. 修改各个节点的主机名和网络配置，分别修改hostname文件和interface文件，依据拓扑结构填写已经规划和分配好的IP地址，子网掩码和DNS服务器及其网关

sudo vim /etc/hostname

sudo vim /etc/network/interface

6. 修改hosts文件，hosts文件存储了主机名和IP地址的映射关系，当发生网络请求时，若请求的网络主机名存在于hosts文件中，则不进行DNS查询，优先使用hosts中的IP地址

sudo vim /etc/hosts

7. 配置SSH，实现无密码登录。SSH可以理解为加密的telnet信道，几点之间可以通过加密的vty通道进行数据访问，控制存储更操作，在进行SSH配置之后，每次数据交互时无需再次输入密码，这在Hadoop中的master节点和slave节点之间具有很重要的意义。

使用 ssh-keygen -t rsa 然后敲击三次回车生成默认的SSH握手文件，打开~/.ssh目录，在master上将公钥放到authorized\_keys中，使用命令：

sudo cat id\_rsa.pus >> authorized\_keys

然后将此authorized\_keys文件拷贝至其他slave节点中的~/.ssh目录下：

sudo scp authorized\_keys hadoop@10.10.11.192:~/.ssh

其中scp的命令格式为 远程主机用户名@远程主机名或IP：存放路径。

修改authorized\_keys文件权限，使文件生效，完成SSH免密登录配置：

chmod 644 authorized\_keys

8. 安装Java环境。由于Hadoop基于Java编写，所以需要使Java环境生效，通过SecureCRT的rz命令将文件上传至各个节点中，将文件放到/usr/lib/java中，解压缩后设置环境变量，追加PATH路径，然后编译生效：

sudo vim ~/.bashrc

export JAVA\_HOME=/usr/lib/java/jdk1.8.0\_40

export PATH=$JAVA\_HOME/bin:$PATH

source ~/.bashrc

1. 上传Hadoop，并配置。还是通过SecureCRT的rz命令，将Hadoop上传至/usr/local/下，解压缩文件并重命名文件

tar -zxvf hadoop1.2.1.tar

sudo mv hadoop1.2.1 hadoop

修改环境变量使hadoop命令生效：

sudo vim ~/.bashrc

export HADOOP\_HOME=/urs/local/hadoop

export PATH=$JAVA\_HOME/bin:$HADOOP\_HOME/bin:$PATH

source ~/.bashrc

分别修改/usr/local/Hadoop/conf下的配置文件，包括hadoop-env.sh/core-site.xml/hdfs-site.xml/mapred-site.xml/master/slave等文件，其目的是对HDFS核心文件配置，MapReduce核心站点配置以及声明master和slave主机。

hadoop-env.sh

export JAVA\_HOME=/usr/lib/java/jdk1.8.0\_40

core-site.xml

<property>

<name>fs.default.name</name>

<value>hdfs://master:9000</value>

</property>

<property>

<name>hadoop.tmp.dir</name>

<value>/usr/local/hadoop/tmp</value>

</property>

hdfs-site.xml

<property>

<name>dfs.replication</name>

<value>2</value>

</property>

<property>

<name>heartbeat.recheckinterval</name>

<value>10</value>

</property>

<property>

<name>dfs.name.dir</name>

<value>/usr/local/hadoop/hdfs/name</value>

</property>

<property>

<name>dfs.data.dir</name>

<value>/usr/local/hadoop/hdfs/data</value>

</property>

mapred-site.xml

<property>

<name>mapred.job.tracker</name>

<value>master:9001</value>

</property>

masters

master

slaves

host1

host2

host3

host4

### 5.1.3 Hadoop集群的启动

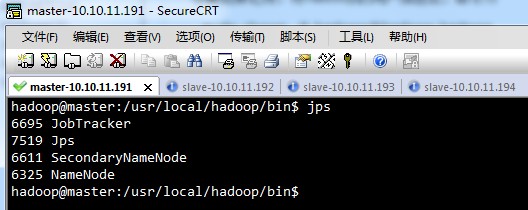
对NameNode进行格式化，仅格式化一次即可

hadoop namenode -format

启动hadoop

cd /usr/local/hadoop/bin

./start-all.sh

通过jps命令可以查看hadoop在各个节点上的运行情况

## 5.2 曲线相似度实验

本文利用天津某供热公司提供的采集数据，对2015-2016供热季中两个供热站的数据进行了分析，依据供热习惯及节能需求，目前很多供热单位晚十点半左右开始维持在某一较低温度，早四五点左右开始升温，即在夜间会将锅炉系统的出水温度维持在较低的水平，所以只对每天6:00至20:00的数据进行分析。由于物联网系统的稳定性等原因会导致缺失值，异常值等离群点的出现，基于时序数据的特点，采用拉格朗日插值法来对数据进行预处理，以此作为实验的初始数据集。

依据（1）式专家根据运行经验确定一天的目标温度调整规律，其中某日的甲锅炉房出水温度曲线及其温度曲线的对比如图2所示。从图中可以看出，目标温度曲线在上午7时左右达到最大值，为一天最冷的时间，供热户在此时间段在家居多，所以需要提供较高热量。随着室外温度、日照等气象条件的提高，出水温度将逐渐降低，而在下午4时开始，供热公司考虑到更多用户将会回到家中，且室外温度和日照逐渐降低，所以升高锅炉出水温度，为用户提供更多热量。对数据预处理后，将甲锅炉房的出水温度曲线和乙锅炉房出水温度曲线分别与目标温度曲线进行一致性分析，依据（2）-（6）式计算出三种属性相似度共五个度量，计算某七天的结果如表1和表2所示。

对数据库中的所有数据进行统计分析，利用最小二乘法并结合锅炉供热系统相关管理人员历史经验，确定（7）式中的复合度量参数值分别为：依据（7）式将五个属性相似度加权融合为一个评价结果，经过对历史数据分析，给出阈值12,42,6,1,3，其一周计算结果如表3和图3所示：

图3可以看出依据本评价方法乙锅炉房的出水温度与目标温度一致性更高，查询原始采集数据可知，甲锅炉房的单位耗气量为10.6，乙锅炉房的单位耗气量为9.2，因此算法是有效的。

由图3可知，第二天和第四天的一致性度量基本相同，进一步分析其原因，对数据按照式（8）进行离差标准化后，直观的对比图如图4和图5所示。

甲锅炉房第二天的纵向伸缩基本一致，趋势变化和纵向平移相较于第四天数据更优，横向平移和横向伸缩相对较差，表现为时间延时较小能及时为用户提供热量，但随目标温度的调控较差。乙锅炉房第二天的趋势变化、纵向平移和纵向伸缩相较于第四天数据更优，而横向平移和横向伸缩相对较差，供热上表现为能为用户提供足够热量，随目标温度合理调控，但其时间延时相对较大。乙锅炉房五个度量皆优于甲锅炉房，供热上表现为乙锅炉房可以按照目标温度进行及时的适度调控，其时间延时较低，相较于甲锅炉房，能在高温时达到供热户需求，低温时能及时降温节约能源。

实际运行中乙锅炉房工作质量较高，根据天气、用户生活习惯、回水温度等各种因素积极调整运行参数，采取自动控制系统来调节出水温度；而甲锅炉房只是按照室外温度进行人工控制，没有精细化供热，其时效性表现较差。所以乙锅炉房的出水温度曲线在与目标温度曲线的一致性上更为接近。两个锅炉房供热户的投诉都很少，则以上数据说明在保证用户室内温度的情况下，乙锅炉房在一定程度上减少了煤气消耗，节约了能源。

## 5.3 聚类并行化实验

## 5.3 实验结论

为了更好地量化管理，达到满足供热的前提下节能减排的目的，本文提出了一个锅炉供热过程的评价方法。对锅炉房的出水温度曲线与目标温度曲线进行一致性分析，分解为趋势变化、平移和伸缩三种属性相似度分析其一致性，给出每个属性相似度的计算方式并加权融合为一个一致性度量。然后利用天津某供热公司的供热数据，对2015-2016供热季的部分数据进行分析，通过本文提出的供热行为评价方法将甲、乙锅炉房进行了对比分析，结果表明，本文提出的评价方法可以较好的区分不同的供热行为，为锅炉供热系统的相关管理人员提供量化考核及其物联网系统参数调整的参考依据，对节能减排，精细化供热具有重大意义。在以后的研究工作将在此基础上，将对实时采集数据进行分析，为及时调整供热行为提供实时建议和参考。

# 供热效果评价实时监测系统

众所周知，数据可视化在数据挖掘的研究过程中扮演着重要的角色，将海量复杂的数据经过数据清洗，聚类分析等过程后得出结果集，通过数值型数据集无法向我们清晰的展示其规律和有趣的知识，为此我们通过数据可视化的手段将数值型数据集表达为图形化数据集，从而让结果一目了然，尤其是在实时系统中，可更快速的发现异常问题，迅速定位故障点从而解决问题，这里将搭建一个供热效果评价实时监测系统，通过实时系统中的采集数据，进行数据同步，进行实时数据分析，并通过前端页面进行展示。

## 6.1 系统架构概述

系统的数据来源源自于已存在的锅炉实时监测系统，通过温度传感器、压力传感器、流量计、气象仪等物联网设备将锅炉系统中实时采集到的数据存放于数据库中。而由于大量的数据是冗余了或者不在我们的分析范围内，且为了保证原系统的稳定性，我们将另外搭建一套数据库服务，这里我们将使用非关系型数据库也就是No SQL数据库MongoDB，后端语言将采用Node.js的中间件Mongoose与后端数据库MongoDB建立连接作为数据访问层，然后将使用Node.js的框架koa方便我们进行敏捷开发，koa被称为下一代web开发框架，具有洋葱模型等分层的概念，这些部分将作为我们后端的系统设计。

由于现在的web开发多为前后端分离模式，且考虑到我们的供热效果评价模型需要较强的前端UI能力，数据渲染能力，所以本系统需要较高的前端性能。为了提高网络中的传输，使用JSON作为标准的通信语言格式，并规范好各个接口的格式，统一规范将大大提高开发效率和协作模式。前端的语言以JavaScript为基础，又将使用最新的ECMA Script规范作为语法，大大丰富了之前ES5的一些窘境。然后使用React作为JavaScript框架进行前端开发，React具有组件化的思想，非MVC类的设计，并使用单向数据绑定让我们分析数据的流向；前端UI组件将使用阿里巴巴系蚂蚁金服开源的ant.design最为UI组件库，其中，我们的CSS将使用SASS这种CSS预处理器进行管理，由于ECMA Script 2015还未完全在各种浏览器中普及，所以我们需要使用Babel对代码进行转义，转义为浏览器可以解析的ECMA Script 5版本，保证系统的稳定性；最后，为了自动化构建系统，方便开发，一体化设计平台的考虑使用webpack对前端代码进行打包，并会包括Babel和SASS编译等功能，还会为图标进行转义成base64格式等，还会使用压缩混淆等功能，从而全方位的提高我们的开发体验，并能保证系统上线后的稳定性及较快的访问速度；并且在对系统开发的过程中会选用Sublime Text 3或者visual studio code等编辑器进行代码书写，保证书写规范，增加开发体验，通过Git来对代码进行版本控制。

## 6.2 技术模块选用

### 6.2.1 MongoDB和Mongoose

MongoDB是一个面向文档的数据库，它并不是关系型数据库，直接存取BSON，这意味着MongoDB更加灵活，因为可以在文档中直接插入数组之类的复杂数据类型，并且文档的key和value不是固定的数据类型和大小，所以开发者在使用MongoDB时无须预定义关系型数据库中的“表”等数据库对象，设计数据库将变得非常方便，可以大大地提升开发进度。在扩展性方面，假设应用数据增长非常迅猛的话，通过不断地添加磁盘容量和内存容量往往是不现实的，而手工的分库分表又会带来非常繁重的工作量和技术复杂度。在扩展性上，MongoDB有非常有效的，现成的解决方案。通过自带的Mongos集群，只需要在适当的时候继续添加Mongo分片，就可以实现程序段自动水平扩展和路由，一方面缓解单个节点的读写压力，另外一方面可有效地均衡磁盘容量的使用情况。整个mongos集群对应用层完全透明，并可完美地做到各个Mongos集群组件的高可用性。这也与我们的MapReduce高度锲合，可以完美的搭配运行起来，同时运用于Linux服务器中。

Mongoose是Node的一个中间件，为了更好的访问MongoDB，同时它为我们封装了诸多常用操作，通过定义Schema、Model和Entity来对一个文档进行抽象，从而可以让开发者通过JavaScript操作BSON数据，进行传统的增删改查操作。

通过下列命令将MongoDB安装在本地服务中：

Mongod --dbpath "D:\MongoDB\data\db" --logpath "D:\MongoDB\data\log\MongoDB.log" --install --serviceName "MongoDB"

### 6.2.2 Node.js和Koa

Node.js是一个运行在服务器端的JavaScript，是JavaScript的一个运行时，Node的出现改变了前端开发的窘境，是前端开发可以扩展到后端甚至移动端，进入大前端时代，一些公司里以Linkendin为代表会将整个后端架构使用Node；而另一部分以阿里巴巴为代表的中途岛计划，将Node作为他们整个架构中的视图渲染层；同时，更多的人会将Node作为前端开发中的工具库，从而丰富整个工程化体系，维护前端开发成本，保证代码质量。我们这里选用Node正因为适应这种Node发展的趋势，又考虑到本供热效果评价系统的业务并不是很复杂，仅仅是数据的读取与前端统计效果的展示。

Koa是基于Node.js平台的下一代web开发框架，koa由Express发展而来，致力于成为一个更小、更富有表现力、更健壮的Web框架。使用koa编写web应用，通过组合不同的generator，可以免除重复繁琐的回调函数嵌套，并极大地提升错误处理的效率。koa不在内核方法中绑定任何中间件，它仅仅提供了一个轻量优雅的函数库，使得编写web应用变得得心应手。本系统将使用koa的洋葱模型做中间件级联：

var koa = require('koa');

var app = koa();

app.use(function \*(next){

var start = new Date;

yield next;

var ms = new Date - start;

this.set('X-Response-Time', ms + 'ms');

});

app.use(function \*(next){

var start = new Date;

yield next;

var ms = new Date - start;

console.log('%s %s - %s', this.method, this.url, ms);

});

### 6.2.3 ECMA Script 2015

ES 2015发布与2015年6月份，又被成为ES 6，是JavaScript语言的下一代标准。它的目标是，是使得JavaScript语言可以用来编写复杂的大型应用程序，成为企业级开发语言。针对于ES 5的诸多糟粕与不足，ES 6进行的更新，通过扩展let/const命令，对var关键词的作用域和生命提升以及全局变量都进行了详细的规范；通过引入箭头函数对之前this作用域问题进行的增强和限制；通过引入import/export关键词对JavaScript语言模块化的不足进行了增强，使我们不必再使用额外的CMD/AMD规范下的模块加载器。本供热效果评价实时监测系统的开发，也是考虑到各大浏览器对ES 6的规范基本支持。

### 6.2.4 React和ant.design

目前较火的前端框架有React/vue/Angular，其中vue和Angular都是MVVM思想的实现，Angular分别发布了1.0和2.0两个版本，是个大而全的解决方案，而vue是一个轻量级小而美的视图层解决方案，采用mobeil first的设计思路。而由Facebook发布的React则是另外一个划时代的JavaScript框架，其采用virtual DOM的思想，利用组件化的设计思路，将前端组件看做一个个有限状态机，当组件状态发生变化的时候执行diff算法，只对DOM变化的部分进行更新，这使得React在前端渲染上相较于其他框架具有得天独厚的优势，尤其在启用前端路由的基础上，页面的跳转将极为迅速。基于本供热效果评价实时监测系统并不具有复杂的业务逻辑和要求较高的视觉效果，且为了将React的组件化设计思想发挥到极致，我们将采用阿里巴巴旗下蚂蚁金服的产品ant.design作为前端UI的组件库，利用其组件将前端界面构建出来

import React from ‘react’;

import ReactDOM from ‘react-dom’;

import 'antd/dist/antd.css';

import { DatePicker } from 'antd';

ReactDOM.render(<DatePicker />, mountNode);

### 6.2.5 SASS、Babel以及Webpack

由于css是一种图灵不完整语言，并不具备编程的元素，只是算作一门配置语言，这样导致了代码逻辑极为混乱，为了解决这个问题，本系统采用SASS这种CSS预处理器，来为CSS加入编程的元素，通过变量和函数定义相应的mixin区块，帮助我们规范的调整代码模块分析CSS代码逻辑，大大节省了我们的开发时间，使代码变得更加简单可依赖。由于之前也使用ES 6这种高级规范，包括SASS在内，在所有的浏览器并未完全支持的情况下我们只能将其算作语法糖，因此我们需要通过Babel进行转义，Babel取自《圣经》中巴别塔的含义，其目的是为了让各自的语言相通，通过使用Babel可以讲ES 6和SASS这种高级规范和预处理语言编译为现代浏览器可以识别的原生内容。

另外，为了方便将系统各个模块的静态资源文件进行管理打包，区分模块之间的依赖关系，同时将Babel集成进流程化的内容中，我们将使用Webpack这种打包工具，它将一些静态资源都视为模块，无论是js代码还是css代码，都可以通过import的方式进行引入，这里也更好的诠释了React组件化的思想，我们将供热效果实时监测系统的各个模块通过依赖关系进行管理，最终打包成为一个bundle.js进行全军引入，大大加快了后续的操作速度，增强了用户体验，以下为webpack的配置文件。

var path = require('path');

var ROOT\_PATH = path.resolve(\_\_dirname);

var APP\_PATH = path.resolve(ROOT\_PATH, 'app');

var BUILD\_PATH = path.resolve(ROOT\_PATH, 'build');

module.exports = {

    entry: APP\_PATH,

    output: {

        path: BUILD\_PATH,

        filename: 'bundle.js'

    },

    module: {

        loaders: [{

            test: /\.jsx?$/,

            loader: 'babel',

            exclude: /node\_modules/,

            query: {

                presets: ['es2015', 'react']

            }

        }]

    },

    resolve: {

        extension: ['', '.js', '.jsx']

    }

};

## 6.3 实时监测展示

# 总结与展望

## 7.1 总结

## 7.2 展望

参考文献

1. 张珊. 供热锅炉绩效评价及优化系统的研究[D]. 大连海事大学, 2013.
2. 路昌海, 刘贵松, 张明琤. 基于支持向量回归的锅炉出水温度时间序列预测[J]. 区域供热, 2014(6):18-22.
3. 岳孝忠. 基于锅炉运行优化的数据挖掘平台研究与实现[D]. 电子科技大学, 2012.
4. 江亿, 彭琛, 胡姗. 中国建筑能耗的分类[J]. 建设科技, 2015(14):22-26.
5. 高兴.基于特征信息的测井曲线相似度算法研究与应用[D]. 东北石油大学, 2013.
6. 张宇, 刘雨东, 计钊. 向量相似度测度方法[J]. 声学技术, 2009, 28(4):532-536.
7. 郑丽萍, 李光耀, 梁永全,等. 本体中概念相似度的计算[J]. 计算机工程与应用, 2006, 42(30):25-27.
8. HELMUT ALT, MICHAEL GODAU. COMPUTING THE FRÉCHET DISTANCE BETWEEN TWO POLYGONAL CURVES[J]. International Journal of Computational Geometry & Applications, 2011, 5(1):75-91.
9. Mosig A, Clausen M. Approximately matching polygonal curves with respect to the Fréchet distance[J]. Computational Geometry, 2005, 30(2): 113-127.
10. 曹凯, 唐进君, 刘汝成. 基于Fréchet距离准则的智能地图匹配算法[J]. 计算机工程与应用, 2007, 43(28):223-226.
11. Eiter T, Mannila H. Computing discrete Fréchet distance. See Also[J]. See Also, 1994, 64(3):636-637.
12. 朱洁, 黄樟灿, 彭晓琳. 基于离散Fréchet距离的判别曲线相似性的算法[J]. 武汉大学学报:理学版, 2009, 55(2):227-232.
13. 蔡启林, 寿晓峰. 供暖热负荷延时曲线及其应用[J]. 区域供热, 1991(2):1-10.
14. 李光, 吴祈宗. 基于结论一致的综合评价数据标准化研究[J]. 数学的实践与认识, 2011, 41(3):72-77.
15. 王福昌, 曹慧荣, 朱红霞. 经典最小二乘与全最小二乘法及其参数估计[J]. 统计与决策, 2009(1):16-17.