改进的并行 fp-growth 算法在工业设备故障诊断中的应用研究

张 斌[1] 滕俊杰[1] 满 毅[2]
(桂林电子科技大学计算机与信息安全学院 广西 桂林 541004)[1]
(北京邮电大学电子工程学院 北京 100876)[2]

摘 要 如今,工业设备不断向智能化、大型化发展,伴随着设备故障日益复杂多样,如何快速、准确地诊断故障成为一个难题。通过研究,提出以大数据技术 Hadoop 为平台,基于兴趣属性列的改进的 fp-growth 算法作为数据挖掘方法,来实现工业设备的故障诊断。实验以工业齿轮箱为例,首先选取两部分数据分别作为训练数据和测试数据,在预处理阶段对训练数据进行空值处理、维度相关性分析以及抽样离散化数据;其次提出基于兴趣属性列的改进的并行 fp-growth 算法,从训练数据中挖掘出属性列与故障之间的关联规则;最后通过测试数据验证关联规则,证明了改进方法的可行性。实验结果表明,基于兴趣属性列改进的并行 fp-growth 算法能够在保证准确率的情况下进行快速故障诊断。

关键词 故障诊断,Hadoop,Fp-growth
中图法分类号 TP277 **文献标识码** A

Application Research of Improved Parallel Fp-growth Algorithm in Fault Diagnosis of Industrial Equipment

ZHANG Bin[1] TENG Jun-jie[1] MAN Yi[2]
(School of Computer and Information Security, Guilin University of Electronic Technology, Guilin, Guangxi 541004, China)[1]
(School of Electronic Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China)[2]

Abstract Nowadays, industrial equipment is becoming more and more intelligent and large-scale. Along with the increasingly complex and diverse equipment failures, how to diagnose faults quickly and accurately has become a challenge. Hence, taking big data technology Hadoop as platform, fp-growth is utilized as big data mining method to realize the fault diagnosis of industrial equipment. Taking the industrial gear box as example, firstly, the two parts of data are selected as the training data and the test data respectively. In preprocessing stage, the training data is processed by null value, the correlation analysis of dimension and discretization of data. Secondly, this paper put forward an improved parallel fp-growth algorithm based on interest to mine the association rules between attribute columns and faults by the training data. Finally, the association rules were verified by the test data to prove the feasibility of the improved method. Experiment results show that the proposed interest based improved parallel fp-growth algorithm can perform fault diagnosis efficiently with accuracy.

Keywords Fault diagnosis, Hadoop, Fp-growth

随着大型智能化工业设备的发展和使用,设备故障变得日益复杂,传统的诊断方法因存在局限性而无法进行快速有效的诊断,人们希望可以从设备监控时产生的异常变化的数据中发现数据与故障间的关系,来准确、快速地发现故障。而在海量的数据中,工作人员难以从中发现有效的故障数据,更无法快速地排查故障。随着大数据技术的发展,在高维的海量大数据中分析和挖掘知识优势,有望解决传统故障诊断难以解决的问题。因此,将大数据技术与故障诊断相结合,利用其强大的数据处理和分析能力,能够很好地解决故障诊断中的难题。

近年来,故障诊断问题受到国内外学者的广泛关注。文献[1]提出了一种基于小波神经网络的故障诊断算法,给出了一种改进的 Laplacian 特征映射算法,通过获得有效的特征信号,并将特征信号输入到构造型小波神经网络中,来得到故障类型。文献[2]提出基于遗传算法的随机森林模型对齿轮故障诊断的研究,通过从振动信号中提取的时间、频率和时间频率域来选择最佳条件参数,在受监督的环境下,利用基于遗传算法的随机森林分类器来实现故障诊断。文献[3]给出了基于 Hadoop 的关联算法 Apriori 对动车组进行故障诊断的大数据解决方案,通过基于 Hadoop 的 Apriori 算法挖掘故障数据间的关联规则,并匹配验证规则以实现故障诊断。综上,虽然小波神经网络与基于遗传算法的随机森林模型能够较好地进行故障诊断,但其更适用于较小数量级的数据,在处理海量大数据时,小波神经网络的网络结构会随着样本集的增大而

本文受国家自然科学基金(61762028),桂林市科学研究与技术开发计划(20160218-1)资助。
张 斌(1970-),男,硕士,副教授,主要研究方向为计算机网络与应用技术、计算机控制技术、网络安全;滕俊杰(1990-),男,硕士生,主要研究方向为大数据挖掘,E-mail:505091074@qq.com(通信作者);满 毅(1975-),男,博士,副教授,主要研究方向为大数据分析与挖掘、网管等。

增大，从而大大降低收敛速度；而基于遗传算法的随机森林模型虽然能够很好地对高维数据进行诊断，但样本数据的增大会使构建森林树模型变得庞大，从而导致处理时间过长或内存不足等问题。而基于 Hadoop 的 Apriori 算法虽然在并行计算处理大数据集时占有优势，但算法自身多次迭代访问事务数据库，生成了大量候选集，降低了算法的整体效率和运行时间。

本文针对处理海量大数据集及 Apriori 算法的不足，提出一种 Hadoop 平台下基于兴趣属性列的 fp-growth 算法。fp-growth 算法将事务数据在内存中以 fp-tree 的方式来存储挖掘，这个过程不会生成候选项集，并且只访问两次数据库。基于 hadoop 并行挖掘与可选择兴趣列输出频繁集的改进，大大提高了算法效率，同时将其应用于工业设备诊断，解决了传统工业设备故障诊断的难题。

1　故障诊断模型的设计

工业齿轮箱故障诊断的架构如图 1 所示，主要包括数据采集、数据存储、数据预处理、数据挖掘和故障诊断 5 个部分。首先，通过传感器等其他数据采集技术采集数据，并存储于数据库；其次，选取一部分数据作为训练数据，将另一部分数据作为测试数据，并对其进行数据预处理，通过空值均值处理和相关性分析对数据进行补全和数据降维；然后，通过 weka 工具对数据进行离散化处理，满足算法挖掘的数据形式的要求；最后，在 Hadoop 平台上对基于 MapReduce 的 fp-growth 算法进行改进，使其可选择兴趣列输出有效频集，在满足用户兴趣度的同时提高挖掘效率。本文以齿轮故障数据作为实验数据，对改进算法进行挖掘，分析实际挖掘效果。

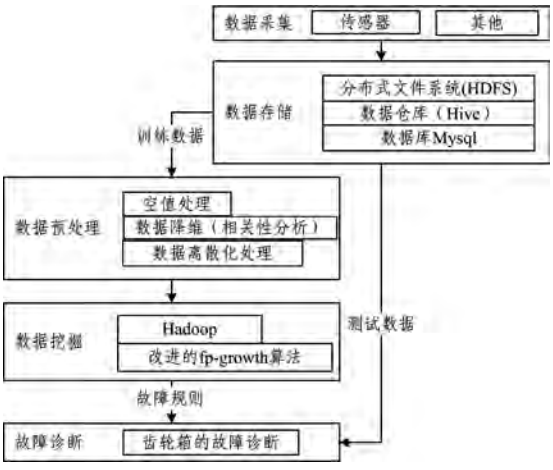

图 1　故障诊断架构

2　数据预处理

2.1　空值处理

由于数据存储的失败、存储器损坏及某些机械因素导致某些数据中存在空值时，为了满足数据的完备性和准确性，本文将采用同类均值方法来填补对应的空值，即空值对应的该属性列相同故障类型的均值。如果缺失值是连续型数据，则以同类中该属性存在值的平均值来填补空值；如果缺失值是离散型数据，则以同类中该属性的众数（即出现频率最高的值）来填补空值。空值处理流程如图 2 所示。

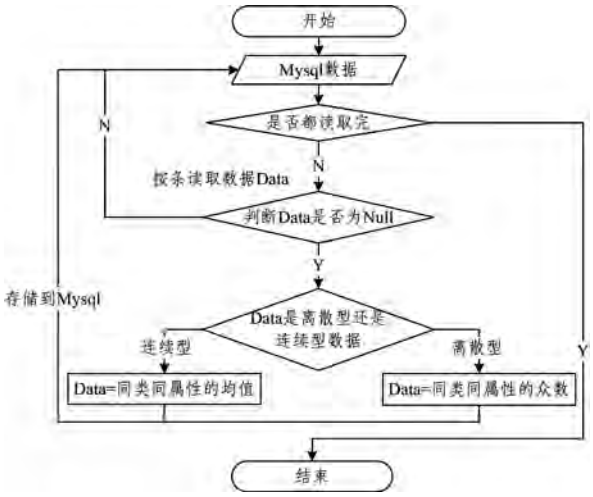

图 2　空值处理流程图

2.2　数据相关性分析

相关系数 ρ_{xy} 的取值范围为[−1,1]，$\rho_{xy}=0$ 时，称 X 和 Y 不相关；$|\rho_{xy}|=1$ 时，称 X 和 Y 完全相关，即 X 和 Y 之间满足 $Y=a+bX$ 的线性函数关系，其中 a 和 b 为常数；$|\rho_{xy}|<1$ 时，X 的变动将引起 Y 的部分变动，ρ_{xy} 的绝对值越大。X 的变动引起 Y 的变动就越大，$|\rho_{xy}|>0.8$ 时为高度相关，当 $|\rho_{xy}|<0.3$ 时，为低度相关，其他为中度相关。

$$\rho_{xy}=\begin{cases}1, & b>0\\ 0, & b=0\\ -1, & b<0\end{cases}$$

相关性系数的计算式如下：

$$\rho=\frac{Cov(X,Y)}{\sqrt{D(X)}\sqrt{D(Y)}}=\frac{\sum(x-\bar{x})(y-\bar{y})}{\sqrt{\sum(x-\bar{x})^2\sum(y-\bar{y})^2}}$$

本文采用相关系数分析，通过相关性程度，在不损伤文本核心信息的情况下剔除高度相关的冗余数据以尽量减少要处理的数据，从而简化计算，提高数据处理的速度和效率。

2.3　数据离散化

由于 fp-growth 算法无法有效挖掘连续型数据，对于离散型数据能更准确地挖掘出数据间的关联关系。此外，有效的离散化数据能够减少算法的计算时间、空间开销及抗噪能力。因此，本文随机抽样故障数据，采用 weka 数据挖掘工具有监督的数据离散化，通过设定类别相关目标函数指标（如分类错误率、熵增益等）对属性数据进行更准确的区间划分。

离散化过程描述：对连续型数据排序 $l=v_0<v_1\cdots<v_n$；选取候选断点 $c_i=(v_{i-1}+v_i)/2$，依次对子集计算信息熵为：

$$H(X)=-\sum_{i=1}^{n}p(x_i)\log_2 p(x_i)$$

计算子集 X 的信息熵与断点 c 条件下的 X 的信息熵的差值，即信息增益。公式如下：

$$G(c)=H(X)-H(X|c)$$

通过计算比较不同断点取值的信息增益来寻找最优断点，从而完成更准确的区间分化。

离散化部分数据划分区间如表 1 所列，由于属性种类较多，我们只列举了部分属性的区间划分及命名，以 tch 属性为例，划分 4 个区间分别为 $(-\infty,-8909.879]$，$(-8909.879,-7613.2235]$，$(-7613.2235,-5499.054]$，$(-5499.054,+\infty)$，分别对其命名为 tch_1，tch_2，tch_3，tch_4。划分区间

的断点分别为 8909.879，－7613.2235，－5499.054，其他属性的划分依次类推。

表 1 离散化部分数据划分区间

属性名	区间名	区间范围	断点
tch	tch_1	(＋∞，－8909.879]	－8909.879
tch	tch_2	(8909.879，－7613.2235]	－7613.2235
tch	tch_3	(－7613.2235，－5499.054]	－5499.054
tch	tch_4	(－5499.054，＋∞)	＋∞
ch1	ch1_1	(－∞，－5411.313]	－5411.313
ch1	ch1_2	(－5411.313，－4269.921]	－4269.921
ch1	ch1_3	(－4269.921，－4151.6615]	－4151.6615
ch1	ch1_4	(－4151.6615，－4134.8765]	－4134.8765
ch1	ch1_5	(－4134.8765，－4123.432]	－4123.432
ch1	ch1_6	(－4123.432，－2398.37355]	－2398.37355
ch1	ch1_7	(－2398.37355，＋∞)	＋∞
…	…	…	…

3 基于兴趣列属性的分布式 fp-growth 算法的实现

3.1 Hadoop

Hadoop[4]是开发和处理大规模数据的软件平台，它实现在大量计算机组成的集群中对海量数据的分布式计算。整个平台最核心的两个部分是 HDFS[5]（Hadoop Distributed File System）和 MapReduce[6]。

HDFS[5]是 Hadoop 的一个分布式文件系统。

HDFS 的结构如图 3 所示，它采用 Master/Slave 主从结构。Namenode 属于 Master 节点，管理数据块映射及名称空间，同时处理 client 的读写请求；Datanodes 属于 Slave 节点，负责存储 client 发送的数据块 Blocks，并执行数据块的读写操作。

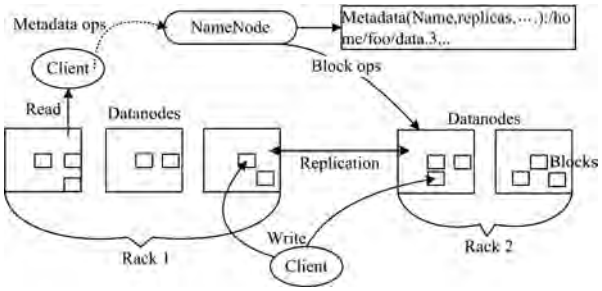

图 3 HDFS 的结构

MapReduce[6]是分布式计算框架，它能自动划分并处理计算数据和计算任务，在集群节点上自动分配和执行任务以及收集计算结果。

MapReduce 的框架如图 4 所示，其定义 Map 和 Reduce 两个抽象编程接口，由用户来编程实现。

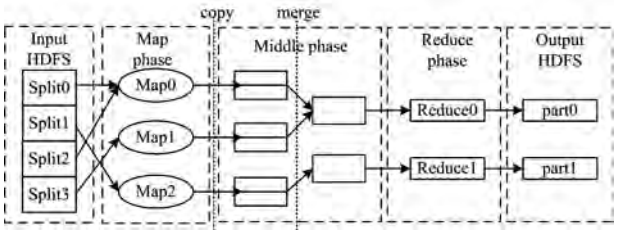

图 4 MapReduce 的框架

Map 过程的输入来自 HDFS 的数据块，通过键值对(key，value)的方式表示数据。Map 函数将处理这些键值对并输出新的键值对(key，value)作为中间数据。

Reduce 过程的输入来自中间过程合并的相同 key 值的键值对，经过进一步的处理，产生最终形式的结果存储到 HDFS。

3.2 fp-growth 关联算法

关联规则挖掘于 20 世纪 90 年代被提出之后迅速成为研究的热点，传统的 Apriori 算法能够很好地挖掘出关联信息，但其缺点是反复扫描数据库并且在获得频繁集模式的同时生成大量的候选频繁集。随后 fp-growth[7]算法被提出以解决该问题。fp-growth 算法采用 fp-tree 的数据结构对事务进行压缩，无需产生候选频集直接得到频繁集，同时其只访问两次数据库，相比 Apriori 算法大大提高了挖掘效率。

一般而言，fp-growth 算法主要分为两个步骤：1)构建 fp-tree；2)从 fp-tree 中挖掘频繁集。

在构建 fp-tree 的过程中，首先扫描一遍数据，统计每一项的频率，删除频率小于最小支持度的项(如最小支持度为 3)。根据项频率由大到小的顺序对数据排序。

表 2 事务数据

TID	项	频数排序项
101	f,a,c,d,g,i,m,p	f,c,a,m,p
102	a,b,c,f,i,m,o	f,c,a,b,m
103	b,f,h,j,o	f,b
104	b,c,k,s,p	c,b,p
105	a,f,c,e,l,p,m,n	f,c,a,m,p

然后，删除不满足最小支持度的项，并按照频数排序，对每一条数据依次添加构造 fp-tree，如图 5 所示。

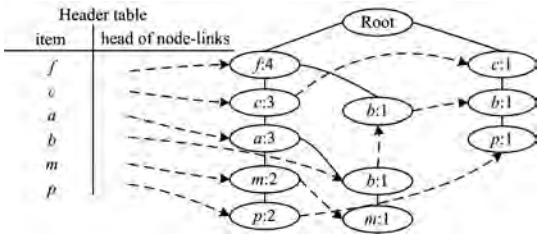

图 5 构建 fp-tree 的过程

在挖掘频繁集的过程中，再扫描一遍数据，分别以表头项作为条件模式基递归挖掘频繁集。以 p 为例，找出所有以 p 为后缀点的分支，有〈f,c,a,m,p〉和〈c,b,p〉两条，以 p 为条件，fp-tree 包含〈f,c,a,m＞和〈c,b〉，然后递归挖掘新的 fp-tree，直到挖掘出所有频繁集组合。

综上可知，fp-growth 算法采用深度优先的处理方法，先后两次遍历事务数据库，把信息压缩到一棵 fp-tree，递归地挖掘频繁项集。由于此算法不产生候选项集，因此相对 Apriori 算法而言，其效率有一个数量级的提高。但是在处理大型数据时，fp-growth 算法需要多次遍历频繁树，构造数据的整体 fp-tree 会非常庞大，会存在内存溢出的可能。同时，挖掘后将输出大量频繁集，在某些实际场合有时仅仅需求某一列或某几列的相关频繁集，用户则需要自己筛选出需求的频繁集，这无意增加了额外的精力和时间。

3.3 改进实现及分析

针对 fp-growth 算法的缺陷，提出基于 MapReduce 对数据进行并行挖掘来减少内存的开销，并在并行计算的同时可选择地输出需求频繁集，这样大大提高了挖掘有用信息的效率。

基于兴趣列属性的分布式 fp-growth 算法的步骤如下：

1)数据分块

将分析的数据上传到 hdfs 文件系统上，然后集群会将数

据分成若干数据块(Block),并将数据存储到集群的各个节点中,用于后续处理。

2)并行计算支持度与兴趣列

通过一次 MapReduce,计算每一个事务项数据的支持度,同时输出支持度表(FList)以及兴趣选择表(CList)。具体地,每个 Map 将从 hdfs 中读取若干个数据块(Blocks),Map 的输入是〈key=事务项,value=1〉,key 表示每一项信息,value 用来计数,每一项的初始化均为 1,同时用"*"标记选择列的项信息。当集群中的所有 Map 处理完数据之后,所有相同的 key 键值对将被分配到同一个 reduce 中,即 reduce 的输入为〈key=事务项,value={1,1,…,1}〉。reduce 对相同 key 的 value 求和,输出为〈key=事务项,value=sum{1,1,…,1}〉。最后,将得到按支持度降序排序的表(FList),同时将标记过"*"的选择列数据存储到表(CList)。

3)FList 分组

为了均衡集群的负载,将 FList 中的项分为 N 组,每组都有唯一的标识(group_id),将所有项以及其所对应的 group_id 以 Hashmap 的形式存储到内存中,形成的表记为 GList。

4)并行挖掘频繁模式树

该过程进行并行化 FP-Growth 的挖掘工作。需要进行一次 MapReduce 计算,即每个 Map 的输入来自第一步分发的事务数据块,Map 的输入是〈key,value=事务数据〉。同时,map 读取第三步生成的 GList 数据表。从事务数据的最后一项开始向前扫描,如果项 An 在 GList 中对应的标识(group_id)是第一次被扫描,则输出$\{A_0,A_1,\cdots,A_n\}$,否则不会输出任何数据。最后将所有相同组别的事务数据放到同一个 reduce 中,reduce 的输入是〈key=group_id,value={{事务数据 1},{事务数据 2},…,{事务数据 N}}〉。每个 reduce 在本地构建 fp-tree,然后递归构建条件 fp-tree,并将挖掘出的频繁模式放入按支持度排序的大根堆中,输出支持度较高的频繁模式〈key=事务项,value={包含该事务项的频繁模式}〉。

5)选择兴趣列输出频繁集

通过一次 MapReduce,挖掘频繁模式中兴趣列的频繁集。将每个事务项作为 key,然后输出包含该 key 的这条频繁模式。每个 Map 中的输出是〈key=事务项,value={该节点上的包含该 item 的频繁模式}〉,在 reduce 过程中,读取第二步生成的选择列的 CList 表中的事务项数据,对所有 Map 的输出结果进行筛选,最终的结果只输出 CList 表中事务项的 key 值包含的频繁集〈key,value={包含该事务项的所有频繁集}〉。

改进后的 fp-growth 算法通过对 MapReduce 的编写完成并行运算,相比传统算法,其运算效率有大幅提高,随着事务数据规模的扩大,这种优势更加明显。筛选兴趣列的改进是指,在第二步并行计算支持度的过程中,通过 Map 前的数据初始化函数 setup 后获取所选兴趣列,在 Map 阶段标记获取兴趣数据列的事务项,在结束 reduce 过程后分别保存支持度列表 FList 和标记的数据列表 CList,并且利用 DistributedCache(MapReduce 框架工具)在节点上执行之前拷贝缓存的 FList 与 CList 路径到 Slave 节点上,用于后续兴趣列数据的挖掘;最后选择输出频繁集的过程中,在初始化函数 setup 中实现读取缓存的 CList 路径来访问得到 CList 的数据,在 Map 阶段只筛选 CList 包含的事务项的键值对〈key=CList 事务项,value=包含的所有频繁集〉,reduce 阶段合并 key 数据并输出选择列的频繁集。

此外,改进后的算法可以根据自身需求对兴趣列进行选择输出,而传统算法无法在挖掘过程中对用户需求的兴趣列进行筛选,造成大量冗余频繁集的输出,因此改进后的算法在满足用户兴趣列需求的同时大幅提高了算法的挖掘效率。

4　实验结果及分析

实验的开发环境包括:1)硬件环境,3 台搭建 Hadoop 集群的 PC 服务器,其中 1 台 32G 内存的主节点,2 台 32G 内存的子节点;2)软件环境,需在 PC 服务器上安装 Linux 版本为 CentOS-6.4-x86_64 的操作系统,以及 java 开发软件 Eclipse 与 1.7 版本的 jdk。

表 3 列出了由 fp-growth 算法在支持度为 0.1、置信度为0.5 时计算得到的故障规则。由表 3 可以看出,如规则为 tch_4,ch1_7,即齿轮箱的转速在 tch_4 即区间$(-5499.054,+\infty)$范围内,输入轴 X 位移在 ch1_7 即区间$(-2398.37355,+\infty)$范围内可推测得出点烛故障;如规则为 tch_2,即转速在 tch_2 即区间$[-8909.879,-7613.2235)$范围内可推测得出断齿故障;如规则为 tch_1,ch1_1,即转速在 tch_1 即区间$(-\infty,-8909.879)$范围内,输入轴 X 位移在 ch1_1 即区间$(-\infty,-5411.313)$范围内可推测得出磨损故障,其他依次类推。

表 3　部分挖掘的关联规则

故障	改进 fp-growth
点烛	tch_4,ch1_7
点烛	ch6_8,ch1_7,tch_4
点烛	ch6_8,ch1_7
断齿	tch_2
断齿	ch1_3,tch_2
磨损	tch_1,ch1_1
磨损	ch5_9,ch1_1,tch_1

表 4 列出了通过挖掘的故障规则对不同条数的记录进行判别的准确率。表 4 中的准确率基本在 80%左右,经过代码计算对错误的记录进行分析时发现,断齿故障判别存在问题,即将正常数据判别为断齿故障,分析得出部分故障规则影响了故障判别,其原因可能是支持度与置信度的选择存在问题,对其重新设置后进行测试。

表 4　故障诊断的准确率

记录条数	正确条数	准确率
1000	802	0.802
2000	1617	0.8085
3000	2425	0.808333333
4000	3228	0.807
5000	4052	0.8104

表 5 列出了反复实验后得到的结果。结果表明在调整的支持度与置信度分别为 0.2 和 0.5 的情况下挖掘出来的规则具有更好的诊断效果。

表 5　重置后的故障诊断准确率

记录条数	正确条数	准确率
1000	1000	1
2000	2000	1
3000	2985	0.995
4000	3977	0.99425
5000	4989	0.9978

算法的运行时间对比如图 6 所示。

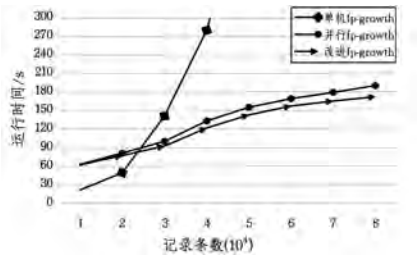

图 6 算法的运行时间对比

由图 6 可得，传统单机 fp-growth 在数据量较少的情况下具有绝对优势，到达一定数量级时计算时间会大幅上涨直到内存不足而无法计算；改进后的 fp-growth 由于基于 hadoop 的并行运算，在处理大数据量时具有一定优势，同时基于兴趣列的选择后减少了频繁集输出的数量，从而较并行 fp-growth 的计算效率有了一定的提升。

结束语 本文在传统 FP-growth 算法的基础上提出了一种改进算法，基于 Hadoop 框架，采用 MapReduce 编程解决了内存溢出的可能，同时实现了对兴趣属性列的筛选。实验结果表明，海量大数据下，减少冗余信息的挖掘，提高了对有效信息的挖掘效率；同时改进后的算法在工业环境设备下的应用有效地解决了传统工业设备故障诊断的难题。

改进的算法虽然解决了大数据故障设备诊断过程中的瓶颈，兴趣度的选择也提高了挖掘效率，但还存在一些问题，fp-growth 算法本身两次访问数据降低了算法效率，针对 hadoop 集群而言，并行计算过程中可能存在负载不均衡问题，而且在工业设备的应用过程中，前期需要足够的监控数据，否则无法构造较好的模型。未来研究将减少算法的访问次数，同时实现 MapReduce 过程中负载均衡的优化，从而将其更好、更准确地应用于工业设备检测。

参 考 文 献

[1] WU L, YAO B, PENG Z, et al. Fault Diagnosis of Roller Bearings Based on a Wavelet Neural Network and Manifold Learning[J]. Applied Sciences, 2017, 7(2): 158.

[2] CERRADA M, ZURITA G, CABRERA D, et al. Fault diagnosis in spur gears based on genetic algorithm and random forest[J]. Mechanical Systems & Signal Processing, 2016, 70-71: 87-103.

[3] 胡辉. 基于 Hadoop 的动车组故障数据关联规则挖掘研究与实现[D]. 北京：北京交通大学，2015.

[4] WHITE T, CUTTING D. Hadoop: the definitive guide[J]. O'reilly Media IncGravenstein Highway North, 2009, 215(11): 1-4.

[5] SHVACHKO K, KUANG H, RADIA S, et al. The Hadoop Distributed File System[C]// MASS Storage Systems and Technologies. IEEE, 2010: 1-10.

[6] DEAN J, GHEMAWAT S. MapReduce: A Flexible Data Processing Tool[J]. Communications of the Acm, 2010, 53(1): 72-77.

[7] AGRAWAL R, SHAFER J C. Parallel Mining of Association Rules[J]. Journal of Supercomputing, 2007, 39(3): 273-299.

[8] LEI Y, JIA F, LIN J, et al. An Intelligent Fault Diagnosis Method Using Unsupervised Feature Learning Towards Mechanical Big Data[J]. IEEE Transactions on Industrial Electronics, 2016, 63(5): 3137-3147.

[9] WANG B, CHEN D, SHI B, et al. Comprehensive Association Rules Mining of Health Examination Data with an Extended FP-Growth Method[J]. Mobile Networks & Applications, 2017, 22(2): 1-8.

[10] 刘鑫，贾云献，孙磊，等. 基于 BP 神经网络的变速箱故障诊断方法研究[J]. 计算机测量与控制，2017，25(1)：12-15.

[11] 谢宏，程浩忠，牛东晓. 基于信息熵的粗糙集连续属性离散化算法[J]. 计算机学报，2005，28(9)：1570-1574.

[12] 杨世海，李涛，陈铭明，等. 基于数据挖掘的智能电网在线故障诊断与分析[J]. 电子设计工程，2017，25(1)：136-139.

[13] LIU D, LIN Y, HUANG P C, et al. Durable and Energy Efficient In-Memory Frequent Pattern Mining[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2017, PP(99): 1-1.

[14] ZENG Y, YIN S, LIU J, et al. Research of improved FP-Growth algorithm in association rules mining [J]. Scientific Programming, 2015, 2015: 6.

[15] 杨鹏坤，彭慧，周晓锋，等. 改进的基于频繁模式树的最大频繁项集挖掘算法——FP-MFIA[J]. 计算机应用，2015，35(3)：775-778.

(上接第 501 页)

人健康状态的发展动向。改善老年人的生活质量与身体状况需要有针对性的工作，本研究对具体的家庭护理提出了针对性建议。

参 考 文 献

[1] 中国老龄科学研究中心课题组. 全国城乡失能老年人状况研究[J]. 残疾人研究，2011(2)：11-16.

[2] 曾毅，顾大男，PURSER J，等. 社会、经济与环境因素对老年健康和死亡的影响--基于中国 22 省份的抽样调查[J]. 中国卫生政策研究，2014，7(6)：53-62.

[3] 张向明. 城市老年人健康状况相关因素初步探讨[J]. 山东医药，2008，48(19)：18.

[4] RONG P, LI L, QUN H. Self-rated health status transition and long-term care need, of the oldest Chinese[J]. Health Policy, 2010, 5(9): 259-266.

[5] 翟德华，陶立群. 高龄老人性格心理特征、饮食习惯与健康长寿关系研究[J]. 中国人口科学，2004，S1(1)：83-87，177.

[6] 谷琳，乔晓春. 我国老年人健康自评影响因素分析[J]. 人口学刊，2006，6(6)：25-29.

[7] 周琼，纪颖，张蕾，等. 高龄老人痴呆发生的相关因素浅析[J]. 中国人口科学，2004，S1(1)：126-129，178.

[8] DEAN A, KOLODY B, WOOD P. Effects of social from various sources on depression in elderly persons[J]. Journal of Health and Social Behavior, 1990, 31(2): 148-161.

[9] LUMV T Y, LIGHTFOOT E. The Effects of Volunteering on the Physical and Mental Health of Older People[J]. Research on Aging, 2005, 27(1): 31-55.

[10] 傅东波，傅华，李洋. 上海市高桥社区糖尿病新型管理模式的效果评估[J]. 中国社会医学杂志，2011，28(6)：412-414.

[11] MERRIL S, BENGTSON V L. Do close parent-child relations reduce the mortality risk of older parents? [J]. Journal of Health and Social Behavior, 1991, 32(4): 382-395.

[12] SCHONE B S, WEINICK R M. Health-related behaviors and the benefits of marriage for the elderly persons[J]. The Gerontologist, 1998, 38(5): 618-627.