

基于 MapReduce 的高维数据频繁项集挖掘

赵欣灿¹, 朱云¹, 毛伊敏²

(1. 江西理工大学 理学院, 江西 赣州 341000; 2. 江西理工大学 信息工程学院, 江西 赣州 341000)

摘要: 传统的数据挖掘算法在面向大规模高维数据的挖掘过程中, 存在数据特征捕捉准确率低、节点负载不均衡、数据交互频繁、频繁项集紧凑化程度低等问题。提出基于 MapReduce 的并行挖掘算法 PARDG-MR, 结合高维数据特征, 设计基于维度粒化算法和负载均衡算法的 DGPL 策略, 并对数据进行预处理, 以解决高维复杂数据特征属性捕捉困难及数据划分中节点负载不均衡的问题。通过构建基于 PJFP-Tree 树的频繁项集并行挖掘策略 PARM, 实现频繁项集的并行化分组过程, 从而提高数据处理的运行效率。在此基础上, 提出基于剪枝前缀推论的整合节点剪枝算法 PJFP, 提高频繁项集挖掘过程中的剪枝效率, 增强频繁项集的紧凑化程度。在 Webdocs、NDC、Gisette 3 个数据集上的实验结果表明, 相比 PFP-growth、PWARM、MRPrePost 算法, 该算法的运行时间平均缩短了约 20%, 能够有效提高数据挖掘效率且降低内存空间。

关键词: 高维数据; 频繁项集; 维度粒化; 并行化; 候选剪枝策略

开放科学(资源服务)标志码(OSID):



中文引用格式: 赵欣灿, 朱云, 毛伊敏. 基于 MapReduce 的高维数据频繁项集挖掘[J]. 计算机工程, 2022, 48(3): 81-89.

英文引用格式: ZHAO X C, ZHU Y, MAO Y M. Frequent itemset mining of high-dimensional data based on MapReduce[J]. Computer Engineering, 2022, 48(3): 81-89.

Frequent Itemset Mining of High-Dimensional Data Based on MapReduce

ZHAO Xincan¹, ZHU Yun¹, MAO Yimin²

(1. School of Science, Jiangxi University of Science and Technology, Ganzhou, Jiangxi 341000, China;

2. School of Information Engineering, Jiangxi University of Science and Technology, Ganzhou, Jiangxi 341000, China)

[Abstract] In the mining process of large-scale high-dimensional data, the traditional data mining algorithm has some problem, such as low accuracy of data feature capture, unbalanced node load, frequent data interaction, and low compactness of frequent itemset. Therefore, this paper proposes a parallel mining algorithm, PARDG-MR which is based on MapReduce. By combining the characteristics of high-dimensional data, a DGPL strategy based on the dimensional granulation algorithm and load balancing algorithm are designed. Data are preprocessed to solve the problems of difficult feature attribute capture of high-dimensional complex data and unbalanced node load in the data division. The parallel grouping process of frequent itemset is realized by constructing a parallel mining strategy PARM of frequent itemset based on the PJFP-Tree to improve the operation efficiency of data processing. On this basis, it proposes an integrated node pruning algorithm PJFP based on the pruning prefix inference, which improves the pruning efficiency in the process of frequent itemset mining to enhance the compactness of frequent itemset and improve the overall mining efficiency of the algorithm. The experimental results on Webdocs, NDC, and Gisette data sets show that compared with PFP-growth, PWARM and MRPrePost algorithms, the running time of PARDG-MR is shorter by approximately 20% on average, therefore, it is more effective and efficient in data mining.

[Key words] high-dimensional data; frequent itemset; dimensional granulation; parallel; candidate pruning strategy

DOI: 10.19678/j.issn.1000-3428.0060253

基金项目: 国家重点研发计划(2018YFC1504705); 国家自然科学基金(41562019); 江西省教育厅科技项目(GJJ151528, GJJ151531)。

作者简介: 赵欣灿(1996—), 女, 硕士研究生, 主研方向为大数据、数据挖掘; 朱云, 副教授、博士; 毛伊敏, 教授、博士。

收稿日期: 2020-12-10 **修回日期:** 2021-03-09 **E-mail:** 2045311239@qq.com

0 概述

关联规则挖掘技术^[1]是数据挖掘领域的重要分支之一,其旨在找出各数据项之间的相互关系,被广泛应用于社交网络、精准营销等领域。然而,随着动态大数据的发展,数据维数逐渐升高。医疗数据、全球气候数据以及多媒体数据等均属于高维数据,且此类数据具有数据量大、数据特征丰富以及数据稀疏等特性^[2]。因此,大部分数据挖掘技术已经从单目标分析扩展为多目标融合分析^[3]。采用高效的方式准确捕捉高维复杂数据的特征属性,以及解决节点分布的负载均衡化问题,并有效提高频繁项集的紧凑程度^[4]成为研究热点。

本文提出基于MapReduce的高维数据频繁项集挖掘算法PARDG-MR。通过设计基于高维数据维度粒化以及节点负载均衡的数据预处理策略,并对复杂的高维数据进行特征属性捕捉,构建基于本地PJFP-Tree树存储结构的算法步骤并行化方法。在此基础上,提出基于剪枝前缀推论(Pruning Prefix Lemma, PPL)的整合节点算法,从而提高算法整体运行速率,且减少算法执行时间和内存。

1 相关工作

高维数据预处理过程包括以特征提取和特征选择^[5]两种方式,在有效消除无关、冗余特征的同时也减小了特征挖掘空间,但是该方法忽略了数据的敏感性以及数据特征之间的关联性。为获得更优的数据特征属性挖掘结果,研究人员对高维数据进行维数约简。文献[6]提出基于图的大数据实体识别算法,并将高维数据关系映射在图中,其中边代表某些数据项间的关系,每条边的权值代表项间关联的程度。该方法避免了重复计算数据项维度属性间的关联程度,并在高维数据的约简^[7]方面取得了相应进展。但是此类方法在数据集中仍存在噪声数据,其是影响数据特征捕捉精确度的关键因素,并且高维数据分布在对应的低维空间中,也会缩短噪声数据与可用特征数据之间的距离,从而降低挖掘数据的价值。因此,在研究高维数据特征约简过程中,粒计算理论^[8]应运而生,其将输入的原始高维数据集分割成包含多个信息粒的小数据集,同时通过粒计算理论对大规模高维数据进行预处理。粒计算理论能够保证数据价值,且精准捕捉数据特征,从而降低数据复杂度。文献[9]提出邻域知识粒度概念用于评估高维数据特征的粒化能力,并将邻域知识粒度与邻域依赖度相结合作为启发式函数,用于数据属性约简。此外,文献[10]针对高维数据系统特征属性的粒度选择问题,分析了信息粒与粒度划分概念,准确地反映决策系统中数据维度粗糙化程度,从而弥补了在大数据环境中高维数据粒化时仅基于论域属性进行约简的不足。文献[11]提出通过对解析式模拟与信息粒进行替换,以定义邻域互补熵、邻域互补条件熵,进而得到非单调性高维数据属性粒化和非单调性高维数据属性约简。因此,通过粒计算理论对高维数据进行预处理,准确捕捉其数据特征成为

挖掘高维数据的研究热点。

高维数据通过粒化降维能在一定程度上提高捕捉数据特征的精确度,但是在大数据环境中数据节点的负载均衡化也一直是研究人员关注的课题。文献[12]提出采用根据节点频繁度不同划分数据的负载均衡策略,通过估计节点的任务数对其进行平均分组,以解决数据倾斜、负载过重的问题。文献[13]提出TBLB算法,该算法结合节点能量和节点度,以形成根据路径性能评价因子进行路径选择的负载平衡树,该平衡树能够有效地均衡节点负载,降低节点能量消耗。目前,大部分负载均衡方法^[14]在分组过程中采用贪心策略,利用计算量模型公式来预估频繁项产生的负载量,将负载量大的项放在负载量总和最小的组以实现负载均衡,然而,计算量模型易造成节点在分布过程中所消耗的时间差异,使得算法的整体效率降低。因此,粒化后的高维数据特征集可以实现分组结果的负载均衡化,成为并行化计算思想发展过程中的主要研究方向。

大数据环境中的FP-growth算法^[15]在并行频繁项集的挖掘执行过程中,数据的频繁交互需要消耗大量资源,使得内存负载压力较大。为此,研究人员尝试将MapReduce数据处理模型与FP-growth挖掘算法相融合,提出了PFP-growth^[16]算法。该算法在执行针对频繁项集的挖掘行为时,每个计算节点无需互相等待或进行节点间的数据交换,均为相互独立的计算节点,从而提高频繁项集在并行挖掘过程中的效率。文献[17]提出MRPrePost算法,在第一次MapReduce任务执行结束后得到频繁1-项集的F-list,并生成PPC-Tree树,对分布在其上的多个计算节点进行频繁项集挖掘,且该过程不需要将PPC-Tree树保存在内存中,既提高了项集支持度的计算效率,又减少了算法在时间与空间方面的消耗。文献[18]提出仅包含单向频繁模式的UFP-Tree树,并引入被约束子树,分别通过递归和非递归方法对指向相同端点和不同端点的路径进行频繁项集挖掘。以上研究旨在减少数据间交互次数且避免消耗过多资源。因此,如何减少因数据频繁交互带来的影响逐渐成为研究热点。

在大数据环境下,数据挖掘效率的影响因素不仅包括数据维度及数据结构,还包括频繁项集紧凑化过程中的算法步骤。近年来,为简化频繁项集的紧凑化步骤,文献[19]提出深度路径优先搜索和长度超集优先检验的改进方法。该方法通过深度优先的路径递归搜索,连续完成最大频繁候选项集的挖掘,并根据路径长度对挖掘结果进行排序,并检验所挖掘的频繁项集超集,以减少候选项集的数量和挖掘次数,从而解决在数据量大、维度高时挖掘效率低的问题。文献[20]提出2FP-Forest算法,该算法通过第一次完全扫描数据集得到全部1-项集和2-项集的支持度,同时充分利用2-项集的剪枝作用,使得所构建树的深度、宽度均比FP-Tree树少。此类算法通过生成最大频繁候选项集以及缩短频繁模式树的构建时间,提高频繁项集的紧凑化程度,从而提升整体的挖掘效率,但是仍无法避免频繁项集在剪枝过程中

的冗余搜索和无效计算。因此,如何在粒化后的高维数据特征集中最大程度提高频繁项集紧凑化程度仍是重要问题。

2 相关概念

定义1 ($N\text{-list}^{[21-22]}$) 给出任意2个 $(k-1)$ -项集 X_A 和 X_B , 使其均为具有相同前缀的频繁项集, 故对应的 $N\text{-list}$ 结构分别表示为:

$$N\text{-list}(X_A) = \left\{ (x_{11}, y_{11}, z_{11}), (x_{12}, y_{12}, z_{12}), \dots, (x_{1m}, y_{1m}, z_{1m}) \right\}$$

$$N\text{-list}(X_B) = \left\{ (x_{21}, y_{21}, z_{21}), (x_{22}, y_{22}, z_{22}), \dots, (x_{2n}, y_{2n}, z_{2n}) \right\}$$

k -项集 X_{AB} 的 $N\text{-list}$ 定义如下:

对于任意 $(x_{1p}, y_{1p}, z_{1p}) \in N\text{-list}(X_A)$, $1 \leq p \leq m$, $(x_{2q}, y_{2q}, z_{2q}) \in N\text{-list}(X_B)$, $1 \leq q \leq n$, 若满足条件 $x_{1p} < x_{2p}, y_{1p} > y_{2p}$, 则将 (x_{1p}, y_{1p}, z_{2q}) 加入到 X_{AB} 的 $N\text{-list}$ 中, 得到初始 $N\text{-list}$ 。

定义2 (JPFP-Tree 树^[23]) 一种树形数据结构, 树中节点由节点名称 (Item-name)、节点计数 (Count)、子节点列表 (Children-list) 组成。

定义3 (空间数据库^[24-25]) 由空间关系数据与对象关系数据集构成, 实现空间数据的数据库化。空间数据库的设计过程包括数据库的逻辑结构设计以及空间数据的集成存储。其中, 数据库的逻辑结构设计采用经典的实体-联系 (Entity-Relation) 图描述现实地理世界, 各图层间的路径数与数据属性特征数成正比, 具体设计如图1所示。

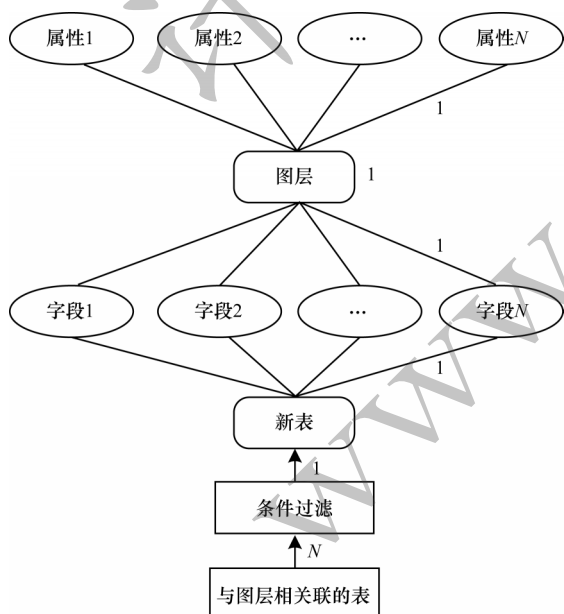


图1 图层间实体-联系模型

Fig.1 Entity-relation model among layers

定义4 (剪枝性质^[26]) 如果项集 $X = \{x_1, x_2, \dots, x_k\}$ 是频繁的, $\forall \alpha \in X \cap \beta \in X \cap \alpha \neq \beta$, 则2-项集 $\{\alpha, \beta\}$ 一定频繁。相反, $\exists \alpha \in X \cap \beta \in X \cap \alpha \neq \beta$, 如果2-项集 $\{\alpha, \beta\}$ 不是频繁项集, 则项集 X 也一定非频繁。

3 PARDG-MR 算法

本文提出的 PARDG-MR 算法的基本原理如下:

1) 提出 DGPL 策略对数据进行预处理, 使得复杂的高维数据完成特征属性的稳定捕捉, 该方法首先提出基于属性精确度计算 (Attribute Accuracy Calculation, AAC) 方法和维度粒大小 (Dimensional Grain Size, DGS) 计算的维度粒化算法 (Dimensional Granulation Algorithm, DGA), 其次根据各特征属性的前缀长度, 提出在粒化后的数据集中基于负载估计 (Load Estimation, LE) 策略的负载均衡算法 (GPL), 提高数据的特征属性捕捉精确度, 从而解决划分中节点负载不均衡的问题。

2) 在数据划分完成的基础上, 设计了本地 PJFP-Tree 树存储结构, 并构建算法步骤并行化策略 (PARM) 来减少数据交互频率, 降低磁盘 I/O 次数, 减缓负载压力。

3) 针对候选剪枝策略, 提出基于剪枝前缀推论 (Pruning Prefix Lemma, PPL) 的整合节点剪枝算法 (PJFP), 该策略将本地生成的 PJFP-Tree 树中的频繁2-项集挖掘结果作为完全剪枝的剪枝条件, 使得 PJFP-Tree 中不存在非潜在候选3-项集, 从而增强频繁项集紧凑化程度, 实现算法整体速率的提高。

3.1 DGPL 策略

大数据环境下的高维数据挖掘算法在进行数据预处理时, 存在以下3个问题:

1) 在数据属性特征捕获过程中属性划分通常仅基于部分相容的数据属性, 而忽略其他数据造成数据的捕获结果不具备全局性。

2) 在数据划分过程中, 无法消除噪声数据, 不能提取单个维度的属性, 使得研究不具有针对性。

3) 分类后的数据集无法将数据均匀分组至各数据节点, 造成数据倾斜、负载不均衡, 使得算法执行效率低。针对以上问题, 本文先对数据集进行特征维度属性粒化处理, 提出 DGA。针对 DGA 算法处理后的数据集, 提出基于 LE 的 GPL 算法, 以实现在数据划分过程中的负载均衡化。

3.1.1 DGA 算法

DGA 算法主要是进行高维数据的预处理, 通过将数据特征属性维度粒化, 以准确地捕捉此类高维数据的特征, 其主要分为2个阶段:

1) 在高维复杂数据集中, 将属性精确度计算方法 AAC 作为设置维度粒大小的参数, 充分挖掘数据的质量, 以避免产生过多噪音数据。

定义5 (AAC 方法) 若数据空间中第 i 维度的数据点个数为 N , 均方差为 S_i , 均值为 μ , 属性精确度为 d_i , 则 AAC 的计算如式(1)所示:

$$d_i = \frac{S_i}{\mu} \quad (1)$$

证明 因为 $S_i = \sqrt{S_i^2} = \sqrt{\sum_{i=1}^n (n_i - \mu)^2}$, 当第 i 维中的属性值较集中时, 即 $n_i \rightarrow \mu$, 则有 $S_i \rightarrow 0$, 故 $\frac{S_i}{\mu} \rightarrow 0$,

即 $\frac{S_i}{\mu}$ 值越小,数据特征属性分布趋于集中,该属性精度越高;反之, $\frac{S_i}{\mu}$ 值越大,数据特征属性分布越离散,该属性精度低,可用性差。此外,当数据在第 i, j 维度中的 S_i 值相同,均值相差较大时,即 $S_1 = S_2, \mu_1 \Rightarrow 0, \mu_2 \Rightarrow \infty$, 则有 $S_1 \Rightarrow \sqrt{\sum_{i=1}^n (n_i)^2}, S_2 \Rightarrow \sqrt{\sum_{i=1}^n (n_i - \infty)^2}$, 当属性值 n_i 发生变化时, $\Delta S_1 \gg \Delta S_2$, 因此 μ 越大, d_i 值越小,所得数据特征属性精度越高,数据扰动率越低;反之,属性精度越低,数据扰动率越高。

2) 基于 AAC 方法根据维度相关性以及总体方差定义的原理,提出维度粒大小计算方法 DGS。该策略定义如下。

定义 6 (DGS 方法) 若数据空间中的总维度数为 D , 各维度的属性精度值为 d_i , 本文选取 d_i 的均值 γ 作为维度粒大小的选取参数,故维度粒大小的计算如式(2)所示:

$$P = D^\gamma \quad (2)$$

证明 因为 $\gamma = \frac{1}{n} \sum_{i=1}^n d_i$, 其中 $0 < d_i \leq 1$, 则 $0 < \gamma \leq 1$, 故 $P < D$, 即 P 所计算出的维度数是根据总体方差及估计精度来确定。该值可用于分析维度的数量以及衡量维度的粒化标准。

以上 2 个阶段实现了用最低的成本完成大量规模数据集粒化的目标,且相比单一的权重选择算法^[27]能较好地捕获特征维度间的相关性。DGA 策略在一定程度上提高了算法的粒化效率,同时减少了数据集的维度数量,从而确保后续挖掘算法的可行性。因此,式(1)和式(2)作为维度粒大小的推理计算公式是合理的。

DGA 算法实例: 设论域 $U = \{x_1, x_2, \dots, x_6\}$ 表示待处理的原始数据集,其中, $A = \{a_1, a_2, a_3, a_4\}$ 为论域 U 上的数据属性,将属性评价分为 1、2、3、4、5 类,分别表示优、良、中、合格、差,如表 1 所示。

表 1 原始数据集信息

Table 1 Information of original dataset

U	a_1	a_2	a_3	a_4
x_1	1	1	1	4
x_2	2	5	1	2
x_3	3	1	1	2
x_4	1	5	1	3
x_5	2	1	1	2
x_6	1	1	1	4

对于 $\forall a_i \in A$, 属性 a_1 包含的维度粒为: $A_1 = \{(a_1, 1), (x_1, x_4, x_6)\}, \{(a_1, 2), (x_2, x_5)\}, \{(a_1, 3), (x_3)\}$ 。属性 a_2 包含的维度粒为: $A_2 = \{(a_2, 1), (x_1, x_3, x_5, x_6)\},$

$\{(a_2, 5), (x_2, x_4)\}$ 。属性 a_3 包含的维度粒为 $A_3 = \{(a_3, 1), (x_1, x_2, x_3, x_4, x_5, x_6)\}$ 。属性 a_4 包含的维度粒为 $A_4 = \{(a_4, 2), (x_2, x_3, x_5)\}, \{(a_4, 3), (x_4)\}, \{(a_4, 4), (x_1, x_6)\}$ 。因此,可得数据集中的数据点个数 $N=6$, 属性 a_1, a_2, a_3, a_4 包含的维度粒均值分别为 $n_1=2, n_2=3, n_3=6, n_4=2, \mu=3.25$, 其均方差 S_i 分别为 1.25、0.25、1.66、1.25, 可得属性精确度 d_i 分别为 0.38、0.08、0.51、0.38。由定义 5 可知,属性精度均值 $\gamma=0.3375$, 总维度数 $D=5$, 维度粒大小 $P \approx 2$ 。因此,经计算后得到的属性 $\{1\}, \{2\}$ 为数据特征属性维度粒化结果。

3.1.2 GPL 算法

通过 DGA 算法完成特征属性维度的精确捕捉后,能够进一步提高分组效率。GPL 算法在粒化后的数据集中执行一次 MapReduce 任务。基于 DGA 算法所得的粒大小,本文根据属性的前缀长度设计了负载估计策略 LE,用于估算该粒所对应的结构路径长度,并进行负载均衡化分组,以确保每组分配到相同的负载量,从而实现整体的负载均衡,负载估计策略 LE 定义如下。

定义 7 (负载估计策略 LE) 已知 k 为节点在 N -list 中的位置, i 为维度粒大小的取整结果,则该节点路径长度的计算如式(3)所示:

$$L(p) = i \times 2^{k-1} \quad (3)$$

证明 对于频繁项 item 而言,假设其在 N -list 中的位置为 k ,其中最差的结果是 item 前任意 $k-1$ 项的组合在该 PJFP-Tree 树中均有相应的路径,该路径中含有 item 项,且此时的路径最多有 2^{k-1} 条,空间数据库中路径数与维度数成正比,因此,式(3)作为路径长度估计函数是合理的。

DGPL 策略的原理是根据 DGA 算法实现高维特征数据的维度粒化,得到属性精确度及数据可分析性均较高的特征属性捕捉结果,从而克服高维数据挖掘算法的局限性。针对粒化后的空间数据集, DGPL 策略通过 GPL 算法对频繁 1-项集进行均匀分组,生成 P -list 列表,实现了节点的均匀分布。

算法 1 DGPL 策略

输入 频繁 1-项集 N -list, 维度数 i , 分组数 G

输出 分组列表 P -list

计算 P -list 中每一项的负载量

1. For each item in N -list do

2. Compute item_load by Eq. (3)

3. P -list = P -list \cup <key = item, value = item.load>

4. End for

5. Sorted(P -list) // 将 P -list 按照 value 值递减顺序排序

6. Return P -list

7. Function getPartition(P -list, G)

8. Return P -list

3.2 PARM 策略

为进一步减少数据库的扫描次数和不必要的计

算消耗,本文提出 PARM 策略以生成本地模式基表树,从而缓解因数据交互次数增多带来的负载压力。虽然 PWARM 算法^[28]省去了 FP-Tree 树和 Head 表的构建过程,但在 Reduce 阶段仍需统计 Map 阶段输出的项集加权支持度。该算法因磁盘 I/O 次数的增加导致计算速度缓慢,从而大幅降低挖掘效率。此外,基于给定项目属性权重的方法,在项目集不断更新过程中无法衡量项目权值,因此,在数据集研究过程中降低挖掘精度且增加了数据集的维护代价。PARM 策略在构造完成 P -list 列表后,映射存储该列表中的频繁 1-项集至本地,再正常调用 MapReduce 过程。

根据 FP-growth 原理,PARM 策略对 F -list 列表中的每个项集进行映射,映射规则为:设频繁 1-项集 $x_1 = \{a\}, x_2 = \{d\}, x_3 = \{b\}, x_4 = \{e\}, x_5 = \{c\}$, 其支持度分别为 7、8、5、5、4。由于列表会重复存储大量的不频繁项,并且读取一个完整事务时需遍历整个列表才能得到目标项集,这样会占用大量的存储空间且消耗大量的时间。因此,根据 FP-growth 原理,本文设计类似 FP-Tree 的树形数据结构 PJFPF-Tree 树。该树在 Reduce 阶段通过与 P -list 列表映射构造得出,此步骤避免统计 Map 阶段输出集合的过程,并相对提高算法的性能。

定义 8(PJFPF-Tree 树) 是具有 $N(N \geq 0)$ 个节点的有限集合,当 $N=0$ 时,称为空树。在任意一棵非空树中应满足以下 2 个条件:1)有且仅有一个名为 null 的节点作为根;2)当 $N>1$ 时,其余每 1 个节点均由频繁 1-项集节点名称(Frequent 1-Itemset)和该节点支持度两部分组成。

该树作为一种逻辑结构,同时也是一种 $M(M=2)$ 层的分层结构,具有以下 2 个特点:1)该树的根节点没有前驱节点;2)树中除根节点外的其他所有节点仅以根节点为其前驱节点,且此类节点均没有后继节点。

生成 PJFPF-Tree 树的目的是在本地磁盘中加快频繁 1-项集的合并速度,使其作为挖掘局部 2-项集的本地模式基,同时大规模减少数据交互频度,从

而缓解负载压力。相比 PWARM 算法的挖掘过程,PARDG-MR 算法中的并行化实现仅基于频繁项集挖掘过程中的 Reduce 阶段,根据分组完成后的 P -list 列表及在频繁项集生成过程中,该阶段构造得到本地频繁模式基 PJFPF-Tree 树,并有效利用分组完成后的数据,避免内存溢出。该过程不存在冗余步骤,为大数据环境下高维数据的挖掘应用提供了优势。

3.3 PJFPF 策略

在大数据背景下生成的 FP-Tree 树规模较大,导致大部分算法忽略了在 FP-Tree 树中进行项集剪枝时带来的内存耗费,从而造成不完全剪枝,且频繁项集紧凑化程度低。因此,在利用 PARM 策略构造完成本地 PJFPF-Tree 树后,本节提出 PJFPF 策略,将 PJFPF-Tree 树中的频繁 2-项集结果作为挖掘局部 2-项集的本地模式基,该策略既考虑了减少再次扫描数据集的时间成本,又使得后续生成的项集中不存在非潜在候选 3-项集。

PJFPF 策略主要分为 2 个步骤:1)提出剪枝前缀推论 PPL,同时利用 PPL 删除可能产生频繁 2-项集的频繁 1-项集,并在后续剪枝过程中将该推论作为 $(k-1)$ 项集阶段的剪枝标准,进而达到后续被整合的节点中不包含非潜在候选 k -项集的目的;2)基于上述挖掘结果,根据频繁模式的支持度对其进行降序排列。

定义 9(剪枝前缀推论 PPL) 如果项集 $\{a, \beta, \gamma\}$ 为非频繁项集,则以其为前缀的所有项集均为非频繁项集。

证明 根据剪枝定理可知, $\exists \alpha \in X \cap \beta \in X \cap \alpha \neq \beta$, 如果 2-项集 $\{a, \beta\}$ 不是频繁项集,则项集 X 也一定是非频繁项集,即若 $\{a, \beta, \gamma\}$ 为非频繁项集,则以项集 $\{a, \beta, \gamma\}$ 为前缀的子集或超集,均是非频繁项集。

为更清晰地描述该推论,本节给出数据集 I, 1-项集和 2-项集的支持度如表 2 所示。

表 2 1-项集和 2-项集的支持度

Table 2 Support degree of 1-item set and 2-item set

元素	A	B	C	D	E	F	G
A	{A,7}	{(A,B),3}	{(A,C),3}	{(A,D),5}	{(A,E),4}	{(A,F),2}	{(A,G),1}
B	—	{B,5}	{(B,C),1}	{(B,D),5}	{(B,E),3}	{(B,F),2}	{(B,G),1}
C	—	—	{C,4}	{(C,D),3}	{(C,E),2}	{(C,F),1}	{(C,G),1}
D	—	—	—	{D,8}	{(D,E),4}	{(D,F),2}	{(D,G),1}
E	—	—	—	—	{E,5}	{(E,F),1}	{(E,G),2}
F	—	—	—	—	—	{F,3}	{(F,G),0}
G	—	—	—	—	—	—	{G,2}

根据表 1 的数据集统计结果,令数据集 I 的支持度 $\min_sup=3$ 。根据本节提出的剪枝推论 PPL 可推

论得出应同时删除项目 F ,其他项目按降序排列,可得结果 $I'=\{D, A, B, E, C\}$ 。执行过程代码如下:

输入 数据集 I , P -list, \min_sup

输出 剪枝后的 P -list

```

1. Foreach  $t \in I$  {
2.  $t' = \text{sort}(t, I')$ 
3. For ( $i = 0$ ;  $i < |I'|$ ;  $i++$ )
4. For ( $j = i + 1$ ;  $j < |I'|$ ;  $j++$ )
5. If ( $D[t_i][t_j] \geq \min\_sup$ )
6. For ( $k = j + 1$ ;  $k < |I'|$ ;  $k++$ )
7. If ( $D[t_i][t_k] \geq \min\_sup$  AND  $D[t_j][t_k] \geq \min\_sup$ )
8. While ( $D[t_{k-1}][t_k] < \min\_sup$ )
9.  $t_{k-1} = t_{k-1}.\text{parent}$ ;
10.  $t_{k-1} = t_k.\text{parent}$ ;
11.  $t_k.\text{count}++$ ;

```

3.4 PARDG-MR 算法

PARDG-MR 算法主要分为以下 4 个步骤。

步骤 1 输入原始数据,通过式(1)和式(2)确定维度粒的大小,完成数据特征属性的精确捕捉。

步骤 2 执行 MapReduce 任务,根据式(3)计算属性前缀长度,实现负载均衡分组,生成包含频繁 1-项集的 P -list 列表。

步骤 3 通过 PARM 方法将 P -list 列表中的频繁 1-项集作为 PJFP-Tree 树的频繁 2-项集模式基。

步骤 4 利用 PJFP 策略实现后续频繁项集的紧凑化,从而完成全局频繁项集的挖掘。

3.5 算法的复杂度分析

3.5.1 时间复杂度分析

PARDG-MR 算法由频繁 1-项集的均匀分组、生成本地频繁模式基 PJFP-Tree 树和频繁项集的并行挖掘过程 3 个部分组成,因此将本文算法 3 个阶段的时间复杂度分别记为 T_{c_1} 、 T_{c_2} 和 T_{c_3} 。

在对高维数据进行特征属性捕捉及获得频繁 1-项集 P -list 的过程中,本文算法需要获得各个记录的数据项中每个维度的属性值,例如每条项集包含的维度数是 N ,记录数是 M ,即在每条项集维度属性值获取过程中的时间复杂度为:

$$T_{c_1} = O(N \times M)$$

在对频繁 1-项集进行均匀分组,生成本地频繁模式基 PJFP-Tree 树时,该步骤仅需在本地完成即可。因此,假设 P -list 的长度为 P ,目标分组产生的数量为 N ,则该阶段的时间复杂度为:

$$T_{c_2} = O(L/G)$$

在频繁项集的并行挖掘过程中,时间复杂度的计算主要包含同一前缀的频繁 $(k-1)$ -项集合并成频繁 k -项集的过程。假设频繁 1-项集 P -list = $\{I_1, I_2, \dots, I_n\}$,以项 I_i 作为结尾的频繁 $(k-1)$ -项集的结构长度为 L_i ,则其时间复杂度为:

$$T_{c_3} = O\left(\sum \sum L_a + L_b\right), a, b \in (0, l)$$

因此, PARDG-MR 算法的时间复杂度为:

$$T_{\text{PARDG-MR}} = O\left(N \times M + \frac{L}{G} + \sum \sum (L_a + L_b)\right)$$

在 PWARM 算法中,第一阶段的时间复杂度基本一致,但在后续分组过程中需要多次扫描数据库,依次比较列表间的元素,可得其时间复杂度为:

$$T_{\text{PWARM}} = O\left((N \times M)^2 + \sum \sum (L_a \times L_b)\right)$$

在大数据环境中, T_{c_1} 、 T_{c_2} 相比于 T_{c_3} 的值较小,甚至可以忽略不计,并且一般情况下的频繁 $(k-1)$ -项集的长度远大于 2,即 $T_{\text{PARDG-MR}} \gg T_{\text{PWARM}}$ 。因此, PARDG-MR 算法的时间复杂度会低于 PWARM 算法。

3.5.2 空间复杂度分析

PARDG-MR 算法的空间复杂度是通过计算频繁项集、 P -list 列表和频繁模式基树存储所需空间的总和。其中,采用模糊属性维度粒化方法使得各个节点的任务量基本相同,使得频繁项集列表结构规模大致相似。假设每个分组结果中平均有 g_i 个频繁 k -项集,其中每个频繁 k -项集均占 b_i Byte,故频繁 k -项集 P -list 的长度均值为 L_k 。频繁 k -项集的每个 P -list 结构占 b_2 Byte,因此 P -list 结构的空间复杂度为 $O\left(\sum_{i=1}^k (g_i \times b_1 + L_i \times b_2)\right)$,然而模式基树存储所需要的空间只由频繁 1-项集决定,其空间复杂度为 $O(b_1 \times L_k!)$,则 PARDG-MR 算法的空间复杂度为:

$$M_{\text{PARDG-MR}} = O\left(\sum_{i=1}^{ik} (g_i \times b_1 + L_i \times b_2) + b_1 \times L_k!\right)$$

PWARM 算法在构建加权 FP-Tree 树时,通常会导致每组节点间存在不均衡负载的问题,且在算法的复杂度评估过程中一般选取最差的结果作为衡量算法复杂度的指标。因此假设分组结果中节点最多有 g_{\max} 个频繁 k -项集,则 PWARM 算法的空间复杂度为:

$$M_{\text{PWARM}} = O\left(\sum_{i=1}^k g_{\max} \times (b_1 + b_2 \times L_i)\right)$$

在大数据背景下,加权 FP-Tree 树的构建远大于 P -list 列表和模式基树所占的存储空间,因此, PARDG-MR 算法的空间复杂度小于 PWARM 算法。

4 实验与结果分析

4.1 实验环境

本文实验硬件方面包含 4 个节点,其中 1 个主节点,3 个辅助节点。4 个节点的 CPU 来自 AMD Ryzen 7,其中 CPU 包含 8 个处理单元,内存均是 16 GB。实验采用的 4 个计算节点都属于同一局域网,均由 200 Mb/s 的以太网互相连接。在软件方面,计算节点安装的均是 2.7.4 版本的 Hadoop 以及 1.8.0 版本的 JDK,其所用的操作系统是 Ubuntu 16.04 版本。各节点的配置如表 3 所示。

表3 实验中各节点的基础配置

Table 3 Basis configuration of each node in the experiment		
主机名	IP地址	角色
Major	192.168.2.101	Major/JobTrace/MajorNode
Sub_1	192.168.2.102	Sub/WorkTrace/DataNode
Sub_2	192.168.2.103	Sub/WorkTrace/DataNode
Sub_3	192.168.2.104	Sub/WorkTrace/DataNode

表4 实验数据集信息

Table 4 Information of experimental datasets		
数据集	样本数	维度数
Gisette数据集	6 000	5 000
NDC数据集	100 000	32
Webdocs数据集	1 692 082	5 267 656

4.2 实验数据集

PARDG-MR 算法使用的 3 个实验数据集均来自标准数据库,分别为 Gisette 数据集、NDC 数据集和 Webdocs 数据集,数据集信息如表 4 所示。其中 Gisette 数据集中包含手写数字识别问题的数据,该数据集样本数量为 6 000,其特征数为 5 000,具有数据量小、特征维度大的特点;NDC 数据集中均为正态分布集群数据,该数据集样本容量为 100 000,维度为 32,具有数据量大、特征维度适中的特点,具有较强的代表性;Webdocs 数据集中包含 1 692 082 条 Web 文档数据,每条数据包含 5 267 656 条不同属性,是一组数据量大、特征维度较大的数据集集合。

4.3 评价指标

为评估 PARDG-MR 算法的运算效率,本文采用加速比作为算法性能的衡量指标。加速比^[29]是指在相同任务量的情况下,使用单一处理器进行运算与使用多个处理器运算所消耗时间的比值。加速比较大,则说明算法在并行计算过程中所消耗的时间越少,挖掘效率越高。

4.4 PARDG-MR 算法可行性分析

本节通过对比算法的加速比,验证 PARDG-MR 算法挖掘执行的可行性。本文分别选取最小支持度阈值为 1 000、3 000 和 5 000,在不同的支持度阈值下将 PARDG-MR 算法分别应用在数据集上,并单独执行 15 次,获得 15 次结果的平均值。在不同数据集上 PARDG-MR 算法的加速比对比如图 2 所示。

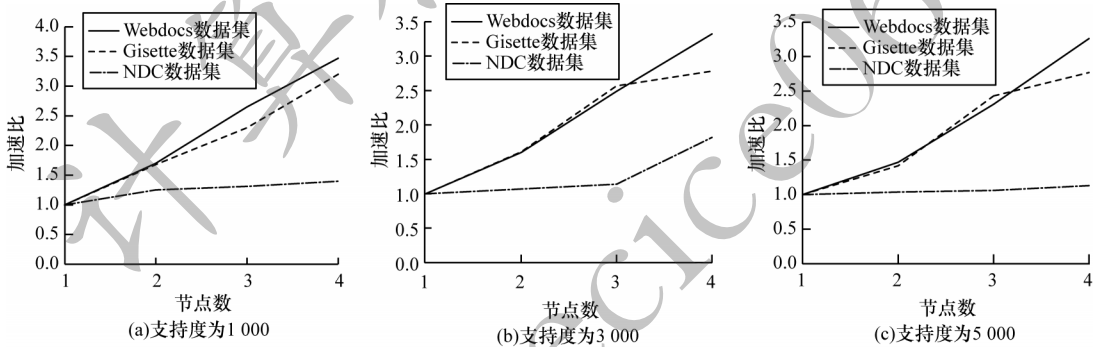


图2 在不同数据集上 PARDG-MR 算法的加速比对比

Fig.2 Acceleration rate comparison of PARDG-MR algorithm on different datasets

从图 2 可以看出,在支持度不小于 3 000 的情况下,PARDG-MR 算法在数据量大、维度数高的数据集环境上执行时,会产生较高的加速比;相反,在类似 NDC 的小数据集中执行时,其加速比增量趋于平稳,而在 Webdocs 的大规模数据集上执行时,随着节点数量的增加,加速比呈上升趋势。当支持度值为 5 000 时,PARDG-MR 算法在同一数据集各节点的加速比相较于该数据集中前一节点的加速比分别提升了 0.91 和 0.99,其原因是在数据量较小、数据维度较低时,现有的数据量不能将集群处理的数据量最大化,此时将数据划分至各节点反而会增大时间开销。然而在大数据环境下,算法优化的关键在于能够完成数据集有效降维以及频繁项集准确合并,在一定程度上能够减少算法在数据集扫描次数,并提高算法整体的挖掘性能。在大规模高维度数据集上,PARDG-MR 算法的挖掘加速比差距接近最大值 1,

说明该算法具有较强的全局挖掘潜力。实验结果表明,PARDG-MR 算法能够适用于大数据背景中高维数据频繁项集的挖掘。

4.5 挖掘性能对比

本文分别在 3 个实验数据集上进行多次对比实验,在算法挖掘过程中将其与 PWARM、PFP-growth、MRPrePost 算法的运行时间分别进行对比。

为验证 PARDG-MR 算法的有效性,本文分别在 Webdocs、Gisette、NDC 这 3 个数据集上将 PARDG-MR、PWARM、PFP-growth、MRPrePost 算法的运行时间进行对比,如图 3 所示。在独立运行 15 次后对其平均值进行分析,通过对比运行时间,评估 PARDG-MR 算法的性能。

从图 3 可以看出,相比 PFP-growth 和 PWARM 算法,PARDG-MR 算法在 3 个数据集上的整体执行时间均有不同程度的减少,其性能表现最佳。在

NDC数据集上 PARDG-MR 算法的运行时间下降幅度最大,相比 Webdocs 数据集和 Gisette 数据集,其运行时间分别下降了 20% 和 64%。在 Gisette 数据集上 PARDG-MR 算法的运行时间下降幅度持续稳定,保持为 6%。相比 PWARM 算法在 NDC 数据集中 3 种支持度下的执行时间, PARDG-MR 算法分别减少了 28%、24% 和 38%。相比 PFP-growth 算法在 Gisette 数据集中 3 种支持度下的执行时间, PARDG-MR 算法分别减少了 28%、17%、15%。执行时间减少的原因在于 PARDG-MR 算法在进行频繁项集挖掘的分组过程中先进行了负载均衡化,并且在构建频繁模式基树时,将冗长的项集结构通过合并方法转化为辨

识度较高的树结构,从而大幅缩短了运算时间。PWARM 算法在获得频繁项集的过程中首先生成了加权 FP-tree 树,同时生成加权节点,该过程消耗了大部分时间;而 PFP-growth 算法在生成频繁项集的过程中需要多次扫描数据集,通过递归的形式构建频繁模式树,从而产生大量的计算开销。由于 PARDG-MR 算法的运行时间在 NDC 数据集上的下降幅度最大,该算法对于数据量大、维度高的数据适应能力最强,其在降低数据维度规模的同时利用频繁模式基结构加速节点的合并,避免了无效计算,在一定程度上提高了算法的挖掘执行效率,且降低了算法的运行时间。

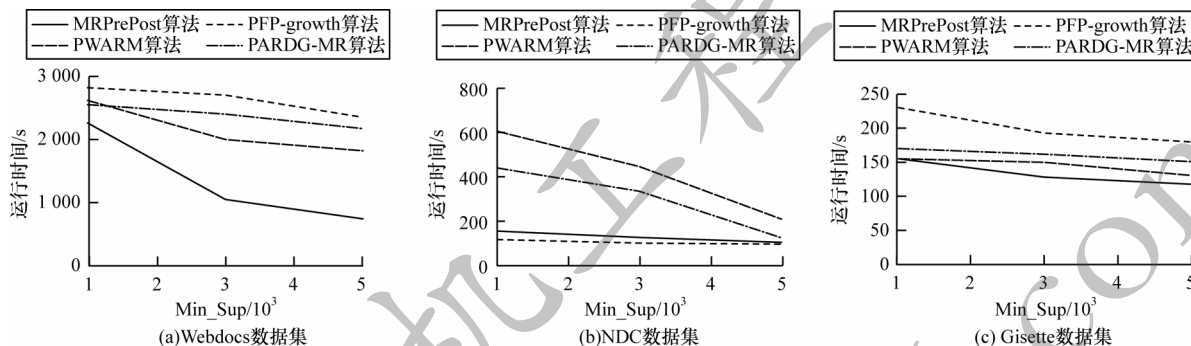


图3 不同算法的执行时间对比

Fig.3 Running time comparison among different algorithms

5 结束语

本文提出基于维度粒化的并行挖掘算法。通过设计 DGPL 策略对高维数据进行预处理,以解决高维数据特征属性捕捉困难以及节点负载不均匀的问题。为进一步提升运行效率,提出 PARM 策略,在本地映射频繁 1-项集列表上构建生成本地 PJFP-Tree 树,以加快节点合并速度、减少数据交互频度并降低负载压力。在此基础上,将频繁 2-项集作为完全剪枝的剪枝条件,构建 PJFP 策略,从而增强频繁项集紧凑化程度,同时降低内存消耗。在 Webdocs、NDC、Gisette 3 个数据集上的实验结果验证了 PARDG-MR 算法的有效性,相比 PFP-growth、PWARM 等算法, PARDG-MR 算法的挖掘效果更佳,能够提高挖掘效率。后续将利用维度间的相关性保留用户更感兴趣的关联规则信息,以提高数据挖掘结果的实用性。

参考文献

[1] AGRAWAL R, SRIKANT R. Fast algorithm for mining association rules[C]//Proceedings of the 20th International Conference on Very Large Data Bases. New York, USA: ACM Press, 1994: 487-499.

[2] 贺玲,蔡益朝,杨征. 高维数据聚类方法综述[J]. 计算机应用研究, 2020, 27(1): 23-31.

HE L, CAI Y Z, YANG Z, et al. Summary of clustering methods for high-dimensional data [J]. Application

Research of Computers, 2020, 27(1): 23-31. (in Chinese)

[3] KAUR M, SINGH D. Multi-objective Bayesian optimization and joint inversion for active sensor fusion[EB/OL]. [2020-11-08]. <https://arxiv.org/abs/2010.05386>.

[4] WANG Z R, WANG H M. Research on mining maximum frequent itemset based on JFP-Growth algorithm [C]//Proceedings of the 2nd International Conference on Mechanical, Electronic and Engineering Technology. Hong Kong, China: [s. n], 2019: 1-10.

[5] LI Z. Implementation of classification and recognition algorithm for text information based on support vector machine[J]. International Journal of Pattern Recognition and Article Intelligence, 2020, 34(8): 1-16.

[6] SAMADI Y, ZBAKH M, TADONKI C. Graph-based model and algorithm for minimizing big data movement in a cloud environment[J]. International Journal of High performance Computing and Networking, 2018, 12(2): 148-155.

[7] 米据生,陈锦坤. 基于图的粗糙集属性约简方法[J]. 西北大学学报, 2019, 49(4): 508-516.

MI J S, CHEN J K. Graph-based approaches for attribute reduction in rough set[J]. Journal of Northwest University, 2019, 49(4): 508-516. (in Chinese)

[8] 冀素琴,石洪波,吕亚丽,等. 基于粒化-融合的海量高维数据特征选择算法[J]. 模式识别与人工智能, 2016, 29(7): 590-597.

Ji S Q, Shi H B, Lv Y L, et al. Feature selection algorithm for massive high-dimensional data based on granulation-fusion[J]. Pattern Recognition and Artificial Intelligence, 2016, 29(7): 590-597. (in Chinese)

[9] 盛魁,董辉,马健,等. 基于邻域粗糙集组合度量的混合数据属性约简算法[J]. 计算机应用与软件, 2020, 37(2):

- 234-239.
SHENG K, DONG H, MA J, et al. Quick reduction algorithm for high-dimensional sets based on neighborhood rough set model[J]. Computer Applications and Software, 2020, 37(2): 234-239. (in Chinese)
- [10] 史进玲,张倩倩,徐久成. 多粒度决策系统属性约简的最优粒度选择[J]. 计算机科学, 2018, 45(2): 152-156.
SHI J L, ZHANG Q Q, XU J C. Optimal granularity selection of attribute reductions in multi-granularity decision system[J]. Computer Science, 2018, 45(2): 152-156. (in Chinese)
- [11] 陈帅,张贤勇,唐玲玉,等. 邻域互补信息度量及其启发式属性约简[J]. 数据采集与处理, 2020, 35(4): 630-641.
CHEN S, ZHANG X Y, TANG L Y, et al. Neighborhood complementary information measurement and its heuristic attribute reduction [J]. Journal of Data Acquisition & Processing, 2020, 35(4): 630-641. (in Chinese)
- [12] VANAHALLI M K, PATIL N. Distributed load balancing frequent colossal closed itemset mining algorithm for high dimensional dataset[J]. Journal of Parallel and Distributed Computing, 2020, 144: 136-152.
- [13] 王潜平,徐琴,王珂,等. 一种基于负载均衡树的多网关节点数据汇集路由算法[J]. 软件学报, 2010, 21: 330-340.
WANG Q P, XU Q, WANG K, et al. Multi-gateway nodes data collected routing algorithm based on load balancing tree [J]. Journal of Software, 2010, 21: 330-340. (in Chinese)
- [14] 杨挺,王萌,张亚健,等. 云计算数据中心HDFS差异性存储节能优化算法[J]. 计算机学报, 2019, 42(4): 47-61.
YANG T, WANG M, ZHANG Y J, et al. Cloud computing data center HDFS differential storage energy saving optimization algorithm [J]. Chinese Journal of Computers, 2019, 42(4): 47-61. (in Chinese)
- [15] 朱颢东,薛校博,李红婵. 海量数据下基于Hadoop的分布式FP-Growth算法[J]. 轻工学报, 2018, 33(5): 36-45.
ZHU H D, XUE X B, LI H C. Distributed FP-growth algorithm based on Hadoop under massive data[J]. Journal of Light Industry, 2018, 33(5): 36-45. (in Chinese)
- [16] 高权,万晓冬. 基于负载均衡的并行FP-Growth算法[J]. 计算机工程, 2019, 45(3): 32-40.
GAO Q, WAN X D. Parallel FP-growth algorithm based on load balancing and redundancy pruning [J]. Computer Engineering, 2019, 45(3): 32-40. (in Chinese)
- [17] LIAO J G, ZHAO Y L, LONG S Q. MRPrePost: a parallel algorithm adapted for mining big data[C]//Proceedings of IEEE Workshop on Electronics, Computer and Applications. Washington D. C., USA: IEEE Press, 2014: 1-10.
- [18] 蒋东洁,李玲娟. 基于单向频繁模式树的频繁项集挖掘算法[J]. 计算机技术与发展, 2019, 29(10): 175-180.
JIANG D J, LI L J. Frequent itemset mining algorithm based on UFP-tree [J]. Computer Technology and Development, 2019, 29(10): 175-180. (in Chinese)
- [19] RAJ S, RAMESH D, SREENU M, et al. EAFIM: efficient apriori-based frequent itemset mining algorithm on spark for big transactional data [J]. Knowledge and Information Systems, 2020, 8(7): 1168-1187.
- [20] 王泽儒. 基于频繁项集挖掘的2FP-Forest算法及其并行化处理研究[D]. 长春: 长春工业大学, 2019.
WANG Z R. Research on 2FP-Forest algorithm based on frequent itemset mining and its parallel processing [D]. Changchun: Changchun University of Technology, 2019. (in Chinese)
- [21] NGUYEN T, NGUYEN L T T, VO B, et al. An N-list-based approach for mining frequent inter-transaction patterns[J]. Special Section on Data Internet of Things, 2020, 8(3): 116840-116855.
- [22] HUYNH V Q P, KUENG J. FPO tree and DP3 algorithm for distributed parallel frequent itemset mining[J]. Expert Systems with Applications, 2020, 140(2): 112874.
- [23] 肖文,胡娟,周晓峰. PFPonCanTree: 一种基于MapReduce的并行频繁模式增量挖掘算法[J]. 计算机工程与科学, 2018, 40(1): 15-23.
XIAO W, HU J, ZHOU X F. PFPonCanTree: a parallel frequent pattern incremental mining algorithm based on MapReduce[J]. Computer Engineering and Science, 2018, 40(1): 15-23. (in Chinese)
- [24] 张自力,秦其明,董开发. 基于ArcSDE的空间数据库设计与实现[J]. 微计算机信息, 2007, 23(11): 133-135.
ZHANG Z L, QIN Q M, DONG K F. The design and implementation of spatial database based on ArcSDE[J]. Microcomputer Information, 2007, 23(11): 133-135. (in Chinese)
- [25] 张学飞,刘锡森,姚薇薇. 一种基于空间数据库的移动网络运营支撑系统[J]. 电信快报, 2020(3): 42-45.
ZHANG X F, LIU X M, YAO W W. A mobile network operation support system based on spatial database [J]. Telecommunications Information, 2020(3): 42-45. (in Chinese)
- [26] WANG Z R, WANG H M. Research on mining maximum frequent itemset based on JFP-Growth algorithm [J]. Electronic and Engineering Technology, 2019: 32-40.
- [27] GUO N, FANG Y, TIAN Z, et al. Research on SOC fuzzy weighted algorithm based on GA-BP neural network and ampere integral method[J]. The Journal of Engineering, 2019, 46(4): 576-580.
- [28] 陈曼如,童向荣,张楠. 基于多尺度粒化的高效正域属性约简算法研究[D]. 烟台: 烟台大学, 2019.
CHEN M R, TONG X R, ZHANG N. Research on efficient positive domain attribute reduction algorithm based on multi-scale granulation [D]. Yantai: YanTai University, 2019. (in Chinese)
- [29] 冯兴杰,潘轩. 基于Spark的投影树频繁项集挖掘算法[J]. 计算机工程与设计, 2018, 39(8): 2477-2483.
FENG X J, PAN X. Projection tree association rule mining algorithm based on Spark [J]. Computer Engineering and Design, 2018, 39(8): 2477-2483. (in Chinese)