(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 110166279 B
(45)授权公告日 2021.05.18

(21)申请号 201910282178.2

(22)申请日 2019.04.09

(65)同一申请的已公布的文献号
申请公布号 CN 110166279 A

(43)申请公布日 2019.08.23

(73)专利权人 中南大学
地址 410083 湖南省长沙市麓山南路932号

(72)发明人 郑美光 杨姣 常成龙 杨柳 胡志刚

(74)专利代理机构 长沙朕扬知识产权代理事务所(普通合伙) 43213
代理人 何湘玲

(51)Int.Cl.
H04L 12/24(2006.01)
H04L 29/08(2006.01)

(56)对比文件
CN 106789350 A,2017.05.31
CN 102013969 A,2011.04.13
WO 2007091123 A3,2008.01.03
US 2016266832 A1,2016.09.15
CN 109565471 A,2019.04.02
胡志刚等.虚拟数据代理云模型构建及数据布局.《中南大学学报》.2019,

审查员 霍远征

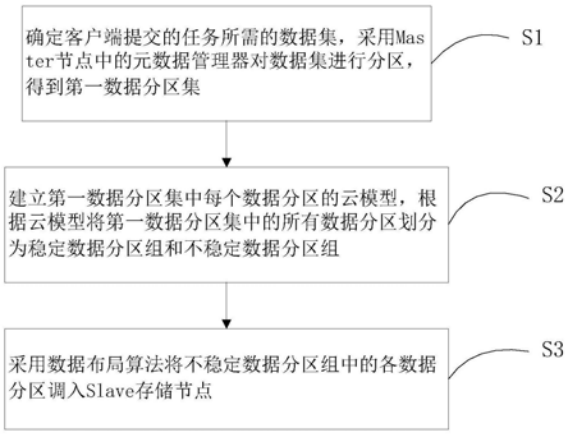
权利要求书2页 说明书10页 附图5页

(54)发明名称

一种非结构化云数据管理系统的动态布局方法

(57)摘要

本发明涉及大数据管理领域,公开了一种非结构化云数据管理系统的动态布局方法,以降低非结构化数据管理系统中由于数据频繁移动导致的数据传输开销,本发明的方法包括确定客户端提交的任务所需的数据集,采用Master节点中的元数据管理器对数据集进行分区,得到第一数据分区集;建立第一数据分区集中每个数据分区的云模型,根据云模型将第一数据分区集中的所有数据分区划分为稳定数据分区组和不稳定数据分区组;采用数据布局算法将不稳定数据分区组中的各数据分区调入Slave存储节点。



CN 110166279 B

1.一种非结构化云数据管理系统的动态布局方法,其特征在于,包括以下步骤:

S1:确定客户端提交的任务所需的数据集,采用Master节点中的元数据管理器对所述数据集进行分区,得到第一数据分区集;

S2:建立所述第一数据分区集中每个数据分区的云模型,根据所述云模型将所述第一数据分区集中的所有数据分区划分为稳定数据分区组和不稳定数据分区组;

S3:采用数据布局算法将所述不稳定数据分区组中的各数据分区调入Slave存储节点。

2.根据权利要求1所述的非结构化云数据管理系统的动态布局方法,其特征在于,所述S3具体包括以下步骤:

S31:设定数据分区dp_j存储在节点s_i上,将$dp_j|_{s_i}$中每个数据集和每个数据集对应的时间传输代价$Cost(D_j[x],dp_j,s_i)$构成dp_j云模型的一个云滴;

S32:将所述云滴输入逆向云发生器,构建该数据分区的云模型,其中,云模型的计算公式为:

$$\begin{cases} Ex=\dfrac{1}{N}\sum_{i=1}^{N}u \\ En=\sqrt{\dfrac{\pi}{2}}\times\dfrac{1}{N}\sum_{i=1}^{N}|u-Ex| \\ He=\sqrt{\left|\dfrac{1}{N-1}\sum_{i=1}^{N}(u-Ex)^2-En^2\right|} \end{cases};$$

式中,Ex表示期望,En表示熵,He表示超熵,N表示一个云模型中云滴的个数,u表示云模型中的每一个云滴;

S33:重复上述S31-S32计算第一数据分区集中所有数据分区的云模型,选取期望值最小的云模型对应的数据分区作为稳定数据分区,选取期望值最大的云模型对应的数据分区作为不稳定数据分区;

S34:计算各云模型对应的数据分区分别与所述稳定数据分区和所述不稳定数据分区的云模型相似度,根据相似度阈值划分得到稳定数据分区组和不稳定数据分区组。

3.根据权利要求2所述的非结构化云数据管理系统的动态布局方法,其特征在于,所述云模型相似度的计算公式为:

$$ECM(DPC_1,DPC_2)=\frac{2S}{\sqrt{2\pi}(En_1+En_2)}\in[0,1];$$

式中,$\sqrt{2\pi}En_1$表示正态云模型DPC_1的期望曲线与横坐标之间形成的面积,$\sqrt{2\pi}En_2$表示正态云模型DPC_2的期望曲线与横坐标之间形成的面积,S表示两个正态云模型的期望曲线相交重叠部分的面积。

4.根据权利要求1所述的非结构化云数据管理系统的动态布局方法,其特征在于,所述Master节点设有查询分析器和数据分区管理器,所述Master节点通过查询分析器分析所述任务的数据处理请求得到任务所需的数据集;且所述Master节点通过所述数据分区管理器调用所述数据布局算法实现动态的数据分区管理。

5.根据权利要求1所述的非结构化云数据管理系统的动态布局方法,其特征在于,所述方法还包括步骤:为每个节点增设监测器;所述S2中,实时收集各数据分区的相关信息通过

所述监测器实现。

6.根据权利要求1所述的非结构化云数据管理系统的动态布局方法,其特征在于,所述S3中,所述采用数据布局算法将所述不稳定数据分区组中的各数据分区调入Slave存储节点具体包括步骤:

将不稳定数据分区组中数据集的本地访问量作为执行复制或迁移的判断依据,当不稳定数据分区组中数据集的本地访问量大于或等于设定阈值时,将不稳定数据分区继续存储至当前节点,执行复制操作;当不稳定数据分区组中数据集的本地访问量小于设定阈值时,执行迁移操作至开销小的其他Slave节点进行存储。

7.根据权利要求1所述的非结构化云数据管理系统的动态布局方法,其特征在于,所述S3中,将所述不稳定数据分区组中的各数据分区调入Slave存储节点时,还包括步骤:复制至少两份所述数据分区,并将复制的各数据分区调入不同的Slave存储节点。

8.根据权利要求1所述的非结构化云数据管理系统的动态布局方法,其特征在于,所述数据布局算法为Random算法或者K-Means算法。

9.根据权利要求1或5所述的非结构化云数据管理系统的动态布局方法,其特征在于,所述各数据分区的相关信息包括数据分区移动数据量的大小、以及被请求次数。

一种非结构化云数据管理系统的动态布局方法

技术领域

[0001] 本发明涉及大数据管理领域,尤其涉及一种非结构化云数据管理系统不稳定分区识别的动态布局方法。

背景技术

[0002] 随着云计算技术、互联网技术以及移动设备技术的发展,人类社会已经进入大数据时代。传统的关系数据库很难满足海量数据的柔性管理需求,近年来,以NoSQL系统为代表的非结构化数据管理系统发展迅速,非结构化数据管理系统作为典型云存储系统,有效地为大数据管理提供支持。大数据环境下,数据的组织管理模式发生很大的改变,数据以分布式形式存储,只有对大数据进行划分才能够真正实现对数据的分布存储,对大数据划分之后,使用合适的策略将数据分区分布存储在各个节点上.所以如何划分和分配数据到节点是NOSQL性能的关键。

[0003] 一些NoSQL数据库直接使用hash分布、随机分布以及轮询分布等传统的分布技术来完成数据分区的分布。例如HBase通过范围划分之后形成的数据分区就使用轮询分布技术分布到系统节点上,Dynamo使用一致性Hash数据划分方法,其实该划分方法蕴含的数据分布策略就是Hash分布技术,有人考虑数据之间的依赖关系,提出一种结合K-means算法的数据布局方案,有效减少了数据移动次数;或者利用数据之间的关系,通过量化计算将它们放置到合适的数据中心,以减少数据中心间的数据移动量;还有人提出根据日志信息挖掘数据关联性,进而将强关联的数据聚合并放置于相同机架上,减少跨机架的数据迁移。以上数据分布技术属于静态分布技术,是指在系统设计之前就设计好的分布技术,并没有考虑系统的操作模式.

[0004] 在云计算环境下部署并执行大数据应用,需要多数据中心的协作.并且用户对大数据的需求是变化的,将会有大量数据需要通过网络进行传输,引起大量的网络传输代价,这就需要动态调整数据分区的存储节点。访问非结构化数据管理系统,高并发的用户查询和修改操作需要对多个数据节点频繁进行I/O操作和数据同步,由高并发访问导致的数据传输开销将导致底层存储系统的各类开销增大。

[0005] 因此,解决非结构化数据管理系统在数据传输管理中的问题至关重要。

发明内容

[0006] 本发明目的在于提供一种非结构化云数据管理系统的动态布局方法,以降低非结构化数据管理系统中由于数据频繁移动导致的数据传输开销,降低系统能耗,提升非结构化数据管理系统的能效水平。

[0007] 为实现上述目的,本发明提供了一种非结构化云数据管理系统不稳定分区识别的动态布局方法,包括以下步骤:

[0008] S1:确定客户端提交的任务所需的数据集,采用Master节点中的元数据管理器对所述数据集进行分区,得到第一数据分区集;

[0009] S2:建立所述第一数据分区集中每个数据分区的云模型,根据所述云模型将所述第一数据分区集中的所有数据分区划分为稳定数据分区组和不稳定数据分区组;

[0010] S3:采用数据布局算法将所述不稳定数据分区组中的各数据分区调入Slave存储节点。

[0011] 优选地,所述S3具体包括以下步骤:

[0012] S31:设定数据分区dp_j存储在节点s_i上,将$dp_j|_{si}$中每个数据集和每个数据集对应的时间传输代价$Cost(D_j[x],dp_j,s_i)$构成dp_j云模型的一个云滴;

[0013] S32:将所述云滴输入逆向云发生器,构建该数据分区的云模型,其中,云模型的计算公式为:

[0014] $$\begin{cases} Ex=\frac{1}{N}\sum_{i=1}^{N}u \\ En=\sqrt{\frac{\pi}{2}}\times\frac{1}{N}\sum_{i=1}^{N}|u-Ex| \\ He=\sqrt{\left|\frac{1}{N-1}\sum_{i=1}^{N}(u-Ex)^2-En^2\right|} \end{cases};$$

[0015] 式中,Ex表示期望,En表示熵,He表示超熵,N表示一个云模型中云滴的个数,u表示云模型中的每一个云滴;

[0016] S33:重复上述S31-S32计算第一数据分区集中所有数据分区的云模型,选取期望值最小的云模型对应的数据分区作为稳定数据分区,选取期望值最大的云模型对应的数据分区作为不稳定数据分区;

[0017] S34:计算各云模型对应的数据分区分别与所述稳定数据分区和所述不稳定数据分区的云模型相似度,根据相似度阈值划分得到稳定数据分区组和不稳定数据分区组。

[0018] 优选地,所述云模型相似度的计算公式为:

[0019] $$ECM(DPC_1,DPC_2)=\frac{2S}{\sqrt{2\pi}(En_1+En_2)}\in[0,1];$$

[0020] 式中,$\sqrt{2\pi}En_1$表示正态云模型DPC_1的期望曲线与横坐标之间形成的面积,$\sqrt{2\pi}En_2$表示正态云模型DPC_2的期望曲线与横坐标之间形成的面积,S表示两个正态云模型的期望曲线相交重叠部分的面积。

[0021] 优选地,所述Master节点设有查询分析器和数据分区管理器,所述Master节点通过查询分析器分析所述任务的数据处理请求得到任务所需的数据集;且所述Master节点通过所述数据分区管理器调用所述数据布局算法实现动态的数据分区管理。

[0022] 优选地,所述方法还包括步骤:为每个节点增设监测器;所述S2中,所述实时收集各数据分区的相关信息通过所述监测器实现。

[0023] 优选地,所述S4中,所述采用数据布局算法将所述不稳定数据分区组中的各数据分区调入Slave存储节点具体包括步骤:

[0024] 将不稳定数据分区组中数据集的本地访问量作为执行复制或迁移的判断依据,当不稳定数据分区组中数据集的本地访问量大于或等于设定阈值时,将不稳定数据分区继续存储至当前节点,执行复制操作;当不稳定数据分区组中数据集的本地访问量小于设定阈值时,执行迁移操作至开销小的其他Slave节点进行存储。

[0025] 优选地，所述S3中，将所述不稳定数据分区组中的各数据分区调入Slave存储节点时，还包括步骤：复制至少两份所述数据分区，并将复制的各数据分区调入不同的Slave存储节点。

[0026] 优选地，所述数据布局算法为Random算法或者K-Means算法。

[0027] 优选地，所述各数据分区的相关信息包括数据分区移动数据量的大小、以及被请求次数。

[0028] 本发明具有以下有益效果：

[0029] 本发明提供的非结构化云数据管理系统的动态布局方法，针对非结构化数据管理系统对存储系统中的数据分区进行云建模，分析数据分区之间的关联关系，识别出不稳定数据分区进行重新布局，可以降低非结构化数据管理系统中由于数据频繁移动导致的数据传输开销，降低系统能耗，可以进一步提升非结构化数据管理系统的能效水平。

[0030] 下面将参照附图，对本发明作进一步详细的说明。

附图说明

[0031] 构成本申请的一部分的附图用来提供对本发明的进一步理解，本发明的示意性实施例及其说明用于解释本发明，并不构成对本发明的不当限定。在附图中：

[0032] 图1是本发明优选实施例的非结构化云数据管理系统不稳定分区识别的动态布局方法流程图；

[0033] 图2是本发明优选实施例的非结构化数据管理系统的数据布局框架；

[0034] 图3是本发明优选实施例第一组实验节点为10个，当任务数量增加时相应的数据传输次数变化情况示意图；

[0035] 图4是本发明优选实施例第一组实验节点为10个，当任务数量增加时相应的数据传输量变化情况示意图；

[0036] 图5是本发明优选实施例第一组实验节点为10个，当任务数量增加时相应的数据传输时间变化情况示意图；

[0037] 图6是是本发明优选实施例第一组实验节点为2-10个，当任务数量增加时相应的数据传输次数增加情况示意图；

[0038] 图7是是本发明优选实施例第一组实验节点为2-10个，当任务数量增加时相应的数据传输量变化情况示意图；

[0039] 图8是是本发明优选实施例第一组实验节点为2-10个，当任务数量增加时相应的数据传输时间变化情况示意图；

[0040] 图9是是是本发明优选实施例第二组实验节点为10个，当任务数量增加时相应的数据传输次数增加情况示意图；

[0041] 图10是本发明优选实施例第二组实验节点为10个，当任务数量增加时相应的数据传输量变化情况示意图；

[0042] 图11是本发明优选实施例第二组实验节点为10个，当任务数量增加时相应的数据传输时间变化情况示意图；

[0043] 图12是本发明优选实施例第二组实验节点为2-10个，当任务数量增加时相应的数据传输次数增加情况示意图；

[0044] 图13是本发明优选实施例第二组实验节点为2-10个,当任务数量增加时相应的数据传输量变化情况示意图;

[0045] 图14是本发明优选实施例第二组实验节点为2-10个,当任务数量增加时相应的数据传输时间变化情况示意图。

具体实施方式

[0046] 以下结合附图对本发明的实施例进行详细说明,但是本发明可以由权利要求限定和覆盖的多种不同方式实施。

[0047] 除非另有定义,下文中所使用的所有专业术语与本领域技术人员通常理解的含义相同。本发明专利申请说明书以及权利要求书中使用的“第一”、“第二”以及类似的词语并不表示任何顺序、数量或者重要性,而仅仅是为了便于对相应零部件进行区别。同样,“一个”或者“一”等类似词语不表示数量限制,而是表示存在至少一个。

[0048] 实施例1

[0049] 参见图1,本实施例提供一种非结构化云数据管理系统的动态布局方法,包括以下步骤:

[0050] S1:确定客户端提交的任务所需的数据集,采用Master节点中的元数据管理器对数据集进行分区,得到第一数据分区集;

[0051] S2:建立第一数据分区集中每个数据分区的云模型,根据云模型将第一数据分区集中的所有数据分区划分为稳定数据分区组和不稳定数据分区组;

[0052] S3:采用数据布局算法将不稳定数据分区组中的各数据分区调入Slave存储节点。

[0053] 上述的非结构化云数据管理系统不稳定分区识别的动态布局方法针对非结构化数据管理系统对存储系统中的数据分区进行云建模,分析数据分区之间的关联关系,识别出不稳定数据分区进行重新布局,可以降低非结构化数据管理系统中由于数据频繁移动导致的数据传输开销,降低系统能耗,可以进一步提升非结构化数据管理系统的能效水平。

[0054] 需要说明的是,非结构化数据管理系统框架中,系统将操作所访问的数据分区分散到多个节点,同时,系统倾向于将任务分配给所需传输数据最少的节点,从其他节点读取本节点没有的数据不可避免地产生大量数据传输。因此,从识别节点中数据传输频繁的数据分区入手,对此类数据分区重新布局,可以减少数据传输开销。本实施例中,在面向NoSQL的数据管理系统中,Slave节点中频繁被动调出数据集而造成数据传输开销大的数据分区称为不稳定数据分区。

[0055] 具体地,本实施例中的非结构化数据管理系统基于以BigTable,Hbase为代表的NoSQL数据管理系统所采用的Master-Slave结构。其非结构化数据管理系统的数据布局框架如下图2所示,在Master-Slave结构中,由Master节点中的元数据管理器对数据集进行分区,再将数据分区分布到不同的数据存储Slave节点上.分区时通常会复制多份副本并将其分布到不同的Slave节点上,既可提高系统的并行性又能避免单点失效.Master节点负责管理整个系统,监视Slave节点的运行状态,以及调度Slave节点的数据分区.所以数据分区的识别工作在主节点实现,作为优选的实施方式,本实施例中的Master节点增设查询分析器(User Request Analyzer,URA)和数据分区管理器(Data Partition Manager,DPM).URA负责分析数据处理请求及其涉及的数据分区.DPM组件负责调用预先存储在策略集中的识别

算法以及布局策略实现动态的数据分区管理.此外,在系统内的各个节点均增设监测器(Monitor),实时收集各节点数据分区移动数据量的大小(数据存储空间的大小)、以及被请求次数等信息。

[0056] 数据位置决定任务的执行位置,计算在数据所在节点上完成。图2中虚线框内容为数据管理系统的工作流程,具体为:客户端将用户提交的任务传送给任务调度管理服务器;任务调度管理服务器根据任务所需数据从Master节点中元数据管理服务器获取数据位置,按照相关任务调度策略将任务调度到Slave节点上;Slave节点根据任务调度管理器的分析结果获取数据并执行任务;任务执行的结果通过任务调度管理服务器传送给客户端,呈现给客户。

[0057] 首先,设目标系统S包括n个Slave节点$S=\{s_1,s_2,...s_n\}$,节点之间的带宽矩阵定义如下:

[0058] $$(b_{ii'})_{nn}=\begin{bmatrix} b_{11} & \cdots & b_{1n} \\ \vdots & & \vdots \\ b_{n1} & \cdots & b_{nn} \end{bmatrix}$$

[0059] 系统中的数据集为$D=\{d_1,d_2,...d_r\}$,每个数据集表示为$d_k=\{size_k\}$,$size_k$表示数据集d_k内所有数据的大小之和。任务集为$T=\{t_1,t_2,...t_l\}$,其中$t_c=\{D_c\}$,D_c表示任务t_c所需的数据集集合,其中,系统的参数说明及定义如下表1所示:

[0060] 表1系统的参数说明

[0061]

参数	描述
s_i	Slave 节点
$b_{ii'}$	节点 s_i 与 $s_{i'}$ 之间的带宽
dp_j	数据分区
D_j	dp_j 中的数据集集合
S_j	dp_j 存储节点集合
$dp_j\|_{s_i}$	存储在 s_i 上的数据分区 dp_j
d_k	数据集
$size_k$	数据集 d_k 大小
$d_k^{i,j}$	存储在 s_i 上的数据分区 dp_j 中的数据集 d_k
$r_i^{d_k^{i,j}}$	节点 s_i 对数据集 $d_k^{i,j}$ 的请求次数
t_c	任务
D_c	任务 t_c 用到的数据集集合

[0062] Master节点把要存储的数据集分成数据分区,某数据分区及其副本被存储在不同的Slave节点上,每个Slave节点将包含多个数据分区。记数据分区集合为$DP=\{dp_1,dp_2,...,dp_m\}$,每个数据分区由二元组$dp_j=\{D_j,S_j\}$表示,其中,$D_j$、$S_j$均为以数组形式存储的集合,$D_j$表示$dp_j$内的数据集集合,通过访问$D_j[x]$可得到对应数据集$d_k$,$S_j$为数据分区及其副本的存储节点集合,通过访问$S_j[y]$可得到数据分区或其副本对应的存储节点$s_i$,如$dp_2=\{D_2,S_2\}->S_2=\{s_1,s_2,s_3,s_7\}$,表示数据分区$dp_2$存储在$s_1,s_2,s_3,s_7$四个节点,$S_2[3]=s_7$,则$dp_2$的其中一个存储节点为$s_7$,$|D_j|$为$dp_j$所含数据集数,$|S_j|$为数据分区副本总数,以下用$dp_j|_{s_i}$表示存储节点为$s_i$的数据分区$dp_j$。

[0063] 将存储在s_i上的数据分区dp_j中数据集d_k记为$d_k^{i,j}$，不同节点对其的请求次数不同，记为$R^{d_k^{i,j}}=\{r_1^{d_k^{i,j}},r_2^{d_k^{i,j}},..r_n^{d_k^{i,j}}\}$。

[0064] 本实施例中，将数据分区对应的被动调出数据集的传输时间作为识别不稳定数据分区的衡量标准，存储在节点s_i上数据分区dp_j中的数据集d_k传输时间总开销计算如下：

[0065] $$Cost(s_i,dp_j,d_k)=\sum_{i=1}^{i-1}\frac{size_k}{b_{i,i'}}\cdot r_i^{d_k^{i,j}}+\sum_{i=i+1}^{n}\frac{size_k}{b_{i,i'}}\cdot r_i^{d_k^{i,j}}$$；

[0066] 首先获得每个数据分区及副本的云模型，云模型作为一种处理定量描述与定性概念的不确定性转换模型，用期望Ex、熵En和超熵He三个数字特征整体表征一个概念的内涵。本实施例中采用云模型理论的逆向云发生器，即根据定量云滴生成定性概念的三个特征值，数据分区dp_j存储在节点s_i上，$dp_j|_{s_i}$中每个数据集和每个数据集对应的时间传输代价$Cost(D_j[x],dp_j,s_i)$即构成dp_j云模型的一个云滴，将云滴输入逆向云发生器，构建数据分区的云模型DPC{Ex,En,He}，为：

[0067] $$\begin{cases} Ex=\frac{1}{N}\sum_{i=1}^{N}u \\ En=\sqrt{\frac{\pi}{2}}\times\frac{1}{N}\sum_{i=1}^{N}|u-Ex| \\ He=\sqrt{\left|\frac{1}{N-1}\sum_{i=1}^{N}(u-Ex)^2-En^2\right|} \end{cases} \quad (2)$$

[0068] 其中，N表示一个云模型中云滴的个数，u表示云模型中的每一个云滴。

[0069] 获得各个数据分区的云模型后，基于每个数据分区的云模型将数据分区分为稳定数据分区组(Stable Group,SG)和不稳定数据分区组(Unstable Group,UG)。对SG和UG进行初始化，选取Ex值最小的数据分区进入SG，Ex值最大的数据分区进入UG，之后，基于云模型相似度的云对比算法，计算每个数据分区与SG和UG的相似度，得到SG及UG内的具体数据分区，从而识别出系统中的不稳定数据分区。

[0070] 对已知的数据分区$dp_j|_{s_j[y]}$，与UG或SG的相似度取其云模型与分组中已有数据分区云模型相似度的最大值作为数据分区$dp_j|_{s_j[y]}$与分组的相似度。本实施例中，两个云模型$DPC_1(Ex_1,En_1,He_1)$和$DPC_2(Ex_2,En_2,He_2)$的相似度计算采用基于期望曲线的云模型相似度(Expectation based Cloud Model,ECM)方法。具体为，通过求解两个云模型的期望曲线相交重叠部分的面积S计算两个云模型的相似度，并根据正向正态云的3En规则，只考虑[Ex-3En,Ex+3En]区间上的云滴分布，求出重叠面积S后，云模型相似度ECM计算如下：

[0071] $$ECM(DPC_1,DPC_2)=\frac{2S}{\sqrt{2\pi}(En_1+En_2)}\in[0,1]$$；

[0072] 式中，$\sqrt{2\pi}En_1$表示正态云模型DPC_1的期望曲线与横坐标之间形成的面积，$\sqrt{2\pi}En_2$表示正态云模型DPC_2的期望曲线与横坐标之间形成的面积，S表示两个正态云模型的期望曲线相交重叠部分的面积。

[0073] 本实施例中，UG内的数据分区即为识别出的不稳定数据分区，算法时间复杂度为O(m·e·σ)，m为系统中数据分区的个数，e表示数据分区中数据集的个数$|D_j|$，σ表示数据分

区的副本个数$|S_j|$,副本数一般取值为3[16]。

[0074] 以3个节点,15个数据集,5个数据分区为例,其初始分布如下表2所示:

[0075] 表2初始数据分布

dp_j	$d_k=\{size_k\}$	s_i	$R_{i,j,k}$
dp_1	$d_1=\{3\}$	s_1	{3,2,6}
	$d_6=\{1\}$		{4,2,2}
	$d_{11}=\{4\}$		{3,1,7}
dp_2	$d_2=\{2\}$	s_2	{4,1,9}
	$d_7=\{5\}$		{6,9,1}
	$d_{12}=\{2\}$		{4,1,9}
		s_3	{3,2,6}
			{2,2,1}
			{8,1,1}
dp_3	$d_3=\{4\}$	s_1	{1,2,2}
	$d_8=\{1\}$		{3,1,1}
	$d_{13}=\{5\}$		{1,2,6}
		s_2	{3,1,3}
			{8,2,6}
			{3,2,6}
dp_4	$d_4=\{1\}$	s_3	{1,5,2}
	$d_9=\{2\}$		{6,2,6}
	$d_{14}=\{1\}$		{3,7,6}
dp_5	$d_5=\{5\}$	s_1	{1,1,5}
	$d_{10}=\{2\}$		{6,8,6}
	$d_{15}=\{3\}$		{7,3,2}

[0076]

[0077] 节点之间的带宽矩阵如下所示:

[0078] $$B=\begin{bmatrix}0 & 10 & 5\\ 10 & 0 & 1\\ 5 & 1 & 0\end{bmatrix}$$

[0079] 由表2得知dp_2及其副本存储在s_2和s_3中,将生成2个云模型,表2中共7个数据分区及其副本,最终得到7个云模型,如表3所示:

[0080] 表3数据分布云模型

[0081]

$dp_j\|_{S_j[y]}$	Ex	En	He
$dp_1\|_{s_1}$	3.6	2.507	1.130
$dp_2\|_{s_2}$	15.2	6.016	1.640
$dp_2\|_{s_3}$	7.467	3.789	1.032
$dp_3\|_{s_1}$	3.233	3.147	1.356
$dp_3\|_{s_2}$	17.167	11.976	4.571
$dp_4\|_{s_3}$	6.4	1.003	0.659
$dp_5\|_{s_1}$	3.867	1.476	0.851

[0082] 由表3可知，七个云模型中，$dp_3|_{s_2}$的Ex值最大(Ex＝17.167)，将其作为UG的初始数据分区。$dp_3|_{s_1}$的Ex值最小(Ex＝3.233)，作为SG的初始数据分区，之后计算其余每个数据分区与SG、UG的相似度，本实施例中的最终识别结果为$SG=[dp_3|_{s_1},dp_1|_{s_1},dp_4|_{s_3},dp_5|_{s_1}]$，$UG=[dp_3|_{s_2},dp_2|_{s_2},dp_2|_{s_3}]$。

[0083] 进一步地，对识别结果UG中的不稳定数据分区，首先确定其操作性质为复制或迁移，将不稳定数据分区中数据集的本地访问情况作为执行复制或迁移的判断依据。当其本地访问量大于或等于设定阈值时，应继续存储至当前节点，执行复制操作，提高任务的并行性；当本地访问量小于设定阈值时，执行迁移操作至开销小的其他节点进行存储。需要说明的是，该处的设定阈值根据具体的云数据管理系统的不同而不同，本发明并不对其阈值的具体取值做限定，本实施例中只对其本地访问量做相对而言的大或者小的比较，并根据其相对比较结果确定执行复制还是迁移操作。进行复制和迁移的区分，可以综合考虑数据分区的本地访问情况，更灵活的实现对不稳定数据的布局。

[0084] 根据当前任务队列收集UG中各数据分区$dp_j|_{S_j[y]}$上数据集的访问信息，使用本地数据时，带宽因素可以忽略不计。本地访问量根据数据大小及其本地访问次数来计算，存储在节点si上，数据分区dp_j中数据集d_k的本地访问量$V(d_k,dp_j,s_i)$为：

[0085] $$V(d_k,dp_j,s_i)=size_k\times r_i^{d_k^{i,j}};$$

[0086] 则数据分区$dp_j|_{S_j[y]}$的本地访问量为：

[0087] $$V(dp_j|_{S_j[y]})=\sum_{x=0}^{|D_j|-1}V(D_j[x],dp_j,S_j[y]);$$

[0088] UG的本地访问量等于所有不稳定数据分区的本地访问量之和，计算如下：

[0089] $$V(UG)=\sum_{dp_j|_{S_j[y]}\in UG}V(dp_j|_{S_j[y]});$$

[0090] UG中的数据分区数记为m^{UG}，不稳定数据分区的平均访问量为：

[0091] $V(UG)_{avg}=V(UG)/m^{UG}$；

[0092] 对UG中的各数据分区$dp_j|_{S_j[y]}$，当$V(dp_j|_{S_j[y]})<V(UG)_{avg}$时，$dp_j|_{S_j[y]}$执行迁移操作，否则，执行复制操作。

[0093] 对于UG中的不稳定数据分区，确定其所需的操作性质为复制或迁移，然后使用现

有数据布局算法将数据分区调入合适的存储节点。

[0094] 实验验证：

[0095] 进一步地，通过建立仿真系统来评估基于不稳定数据分区识别的动态布局策略的效率和性能，本实施例中进行两组实验:第一组使用随机算法(Random)作为数据布局算法，使用本发明的方法(CM-UPI算法)识别出不稳定数据分区之后采用Random布局(CM-UPI+Random)与在初始数据分配的基础上直接采用Random算法布局数据分区减少的数据传输开销进行对比，但需要说明的是，不稳定分区识别算法会产生新的开销，当CM-UPI+Random所减少的传输开销不超过其布局带来的开销时，进行不稳定识别将得不偿失，为此，在进行实验验证时，同时考虑了CM-UPI+Random-Layout(在进行第二组实验时，为CM-UPI+K-means-Layout)的方法，并对比各方法布局过程中产生的传输开销如下。本实施例的附图中，Random是采用Random数据布局后对初始数据分配减少的数据传输开销，Random-Layout是Random数据布局过程中产生的传输开销，CM-UPI+Random是使用本发明的方法(CM-UPI算法)识别出不稳定数据分区之后采用Random布局对初始数据分配减少的数据传输开销，CM-UPI+Random-Layout是使用本发明的方法(CM-UPI算法)识别出不稳定数据分区之后采用Random布局过程中产生的传输开销。

[0096] 第二组实验采用K-means算法作为布局算法，实验中存储的数据集为3000个，通过动态调节任务及节点的数量验证CM-UPI算法的识别效果。

[0097] 本实施例中进行仿真实验的物理平台为Intel core i5-6200U CPU,4GB内存的PC机，实验过程中对节点、任务、数据集的仿真参数设置为：节点之间的带宽值为100～300，单位为MB/s，任务所需数据集个数服从正态分布，均值设为70，方差设为15，数据集大小服从正态分布，均值设为60 MB，方差设为20。

[0098] 进行第一组实验具体包括，设置节点为10个，任务数量从400～2000，当任务数量增加时相应的数据传输次数如图3所示，传输量如图4所示，传输时间的变化如图5所示。

[0099] 由图3-图5可知，随着任务数量的增加，CM-UPI+Random算法减少的传输次数、传输量、传输时间持续增长，并且远远超过其布局开销，同时，CM-UPI+Random所减少的数据传输开销远大于直接采用Random布局数据分区所减少的传输开销，直接用Random布局时不仅有较高的布局开销，而且在数据传输方面没有优化效果。同时，两种算法对应的布局开销随着任务数量的增加均未有明显的涨幅，Random的布局开销远远超过CM-UPI+Random的布局开销。

[0100] 实验过程中，随着任务数量增加，对数据传输的需求也相应的增加，Random算法以随机生成数据分区存储节点的方式布局，使用Random对当前数据分区进行布局，系统内的数据分区都可能被移动，从而产生高额的布局开销，同时会使存储节点较优的数据分区被布局到传输频繁的节点，使得数据传输得不到优化，通过CM-UPI算法识别出来的数据分区，存储在传输频繁的节点，即使使用Random布局也是将数据分区布局到传输开销更少的节点，同时CM-UPI+Random算法只针对不稳定数据分区，具有较低的布局开销。

[0101] 进一步地，设置任务数量为2000，节点数量从2～10，当节点数量增加时相应的数据传输次数如图6所示，传输量如图7所示，传输时间的变化如图8所示。

[0102] 由图6-图8可知，随着节点数量的增加，Random算法的布局开销呈持续增长的趋势，CM-UPI+Random算法的布局开销趋于稳定并且小于Random算法的布局开销。同时，随着

节点数量的增加,CM-UPI+Random算法减少的传输次数、传输量、传输时间开销远远超过其布局开销,并且优于Random。在用Random布局数据分区时不仅有较高的布局开销,而且在数据传输方面没有优化效果。

[0103] 实验过程中,与随着任务数量增加的情况相似,不同的是,随着节点数量的增加,CM-UPI+Random算法减少的传输次数、传输量、传输时间先增加后减少,其原因是节点较少时,Random算法可选择的节点少,分布到较优节点的概率大,随着节点的增加,数据分区被分布到较优节点的概率逐渐降低,从而优化效果减弱。但依然是CM-UPI+Random算法优化的传输次数、传输量、传输时间远远超过其布局开销,并且优化效果远超Random算法,再次验证CM-UPI算法的有效性。

[0104] 进行第二组实验具体包括:设置节点为10个,任务数量从400～2000,当任务数量增加时相应的数据传输次数如图9所示,传输量如图10所示,传输时间的变化如图11所示。

[0105] 由图9-11可知,随着任务数量的增加,CM-UPI+K-means算法减少的传输次数、传输量、传输时间持续增长,远远超过其布局开销.并且在其布局开销小于K-means布局开销的情况下,优化效果逐渐优于K-means。同时,随着任务数量的增加,两种算法对应的布局开销均趋于稳定,K-means的布局开销约是CM-UPI+K-means布局开销的两倍。

[0106] 实验过程中,随着任务数量增加,对数据传输的需求也相应的增加,K-means算法以传输次数为标准,将支持任务相似的数据分区聚类到同一节点,用该节点调度其支持度最大的任务,有效的减少了数据传输次数,使得传输量以及传输时间也得到相应的降低。只使用K-means算法布局时,所有依赖度最大的数据分区聚类到同一节点,存储节点较优的数据分区迁移到其依赖度最大的节点,使部分数据分区的布局开销大于减少的开销,CM-UPI算法识别出来的数据分区存储在传输频繁的节点,在此基础上使用K-means算法布局,将数据分区放置到开销小的节点,布局开销相对于K-means布局时少。同时CM-UPI算法将部分不稳定数据分区进行复制,提高数据的复用,使得算法有更好的优化效果。

[0107] 进一步地,设置任务数量为2000,节点数量从2～10,当任务数量增加时相应的数据传输次数如图12所示,传输量如图13所示,传输时间的变化如图14所示。

[0108] 由图12-14可知,与随着任务数量增加的情况相似,随着节点数量的增加,CM-UPI+K-means算法减少的传输次数、传输量、传输时间持续增长,远超过其布局开销。并且在其布局开销逐渐小于K-means布局开销的情况下,优化效果优于K-means。同时,随着节点数量的增加,K-means算法的布局开销呈持续增长的趋势,CM-UPI+K-means算法的布局开销趋于稳定并且小于K-means算法的布局开销。

[0109] 由此可见,本发明的方法(CM-UPI算法)识别出来的不稳定数据分区的重新布局可以有效改善数据传输的开销,验证了识别不稳定数据分区算法的有效性。

[0110] 实施例2

[0111] 与上述方法实施例相对应地,本实施例提供一种非结构化云数据管理系统不稳定分区识别系统,包括存储器、处理器以及存储在存储器上并可在处理器上运行的计算机程序,所述处理器执行所述计算机程序时实现上述方法的步骤。

[0112] 以上所述仅为本发明的优选实施例而已,并不用于限制本发明,对于本领域的技术人员来说,本发明可以有各种更改和变化。凡在本发明的精神和原则之内,所作的任何修改、等同替换、改进等,均应包含在本发明的保护范围之内。

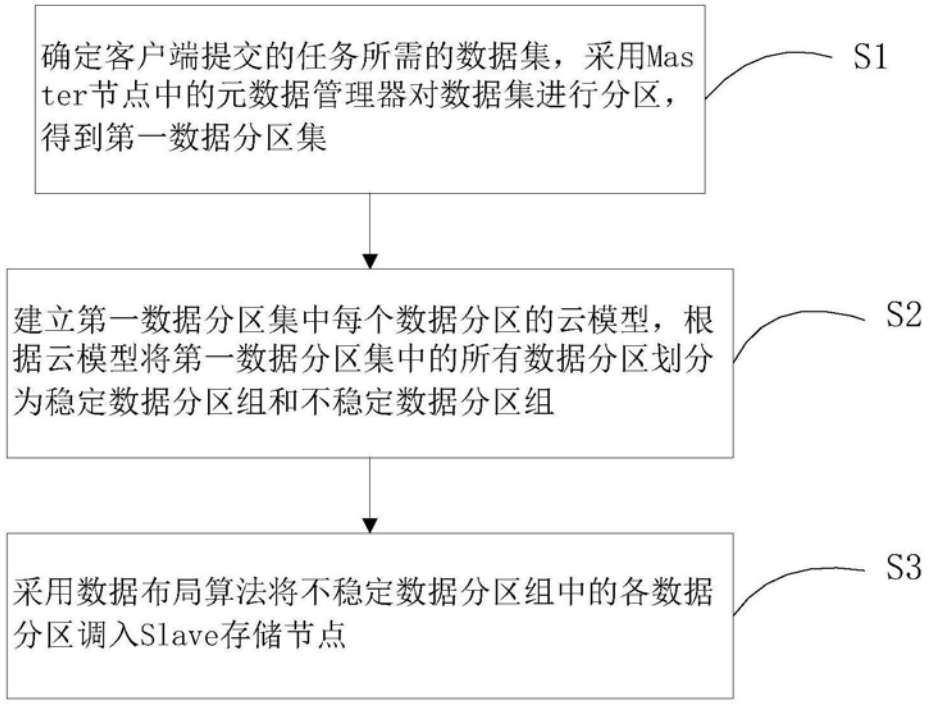

确定客户端提交的任务所需的数据集，采用Master节点中的元数据管理器对数据集进行分区，得到第一数据分区集
S1
建立第一数据分区集中每个数据分区的云模型，根据云模型将第一数据分区集中的所有数据分区划分为稳定数据分区组和不稳定数据分区组
S2
采用数据布局算法将不稳定数据分区组中的各数据分区调入Slave存储节点
S3

图1

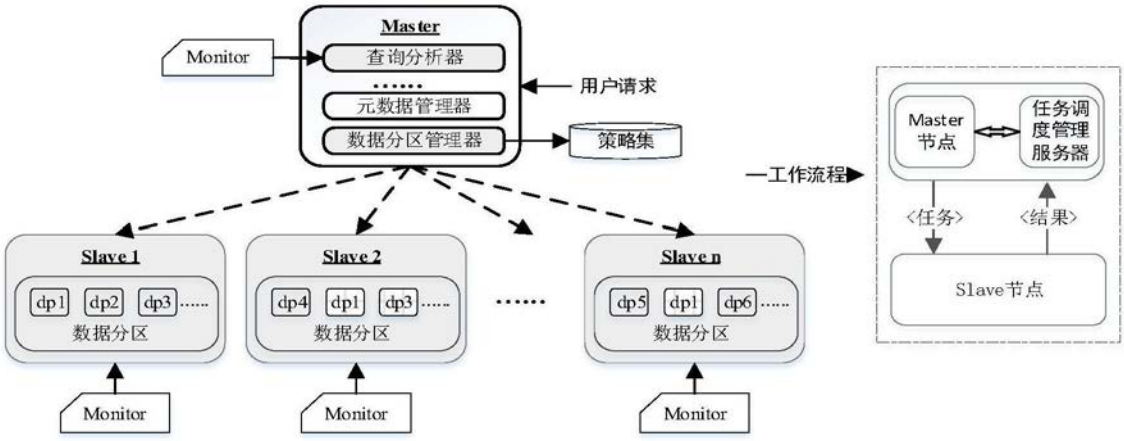

Master
Monitor
查询分析器
元数据管理器
数据分区管理器
用户请求
策略集
工作流程
Master节点
任务调度管理服务器
<任务>
<结果>
Slave节点
Slave 1
Slave 2
Slave n
dp1 dp2 dp3 ..... 数据分区
dp4 dp1 dp3 ..... 数据分区
dp5 dp1 dp6 ..... 数据分区
Monitor
Monitor
Monitor

图2

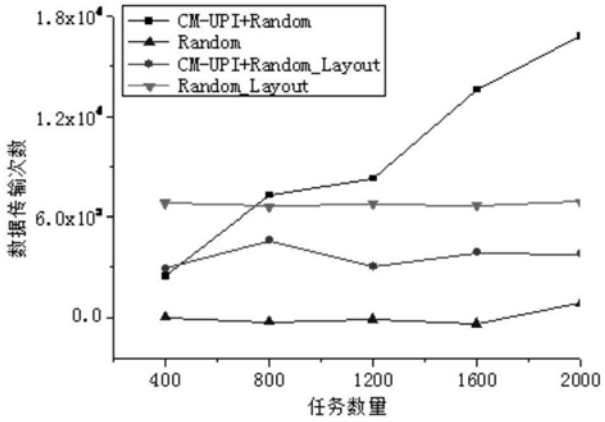

CM-UPI+Random
Random
CM-UPI+Random_Layout
Random_Layout
数据传输次数
任务数量
1.8x10⁴
1.2x10⁴
6.0x10³
0.0
400
800
1200
1600
2000

图3

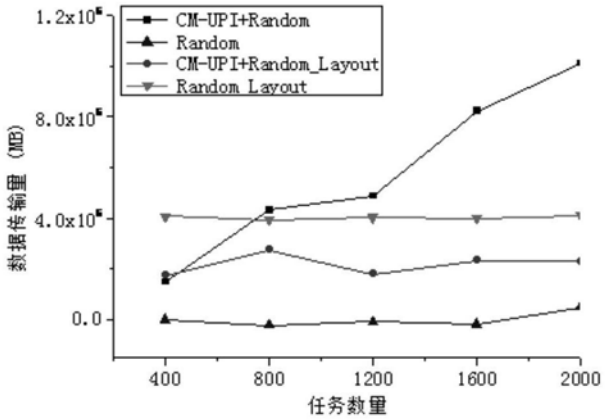

CM-UPI+Random
Random
CM-UPI+Random_Layout
Random_Layout
数据传输量 (MB)
任务数量
1.2x10⁶
8.0x10⁵
4.0x10⁵
0.0
400
800
1200
1600
2000

图4

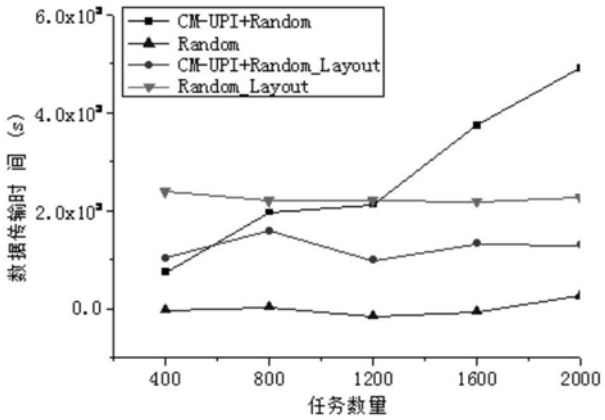

CM-UPI+Random
Random
CM-UPI+Random_Layout
Random_Layout
数据传输时间 (s)
任务数量
6.0x10³
4.0x10³
2.0x10³
0.0
400
800
1200
1600
2000

图5

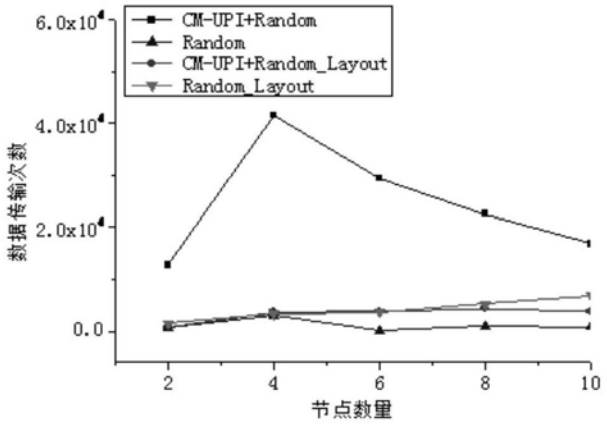

CM-UPI+Random
Random
CM-UPI+Random_Layout
Random_Layout
6.0x10⁴
4.0x10⁴
2.0x10⁴
0.0
数据传输次数
2
4
6
8
10
节点数量

图6

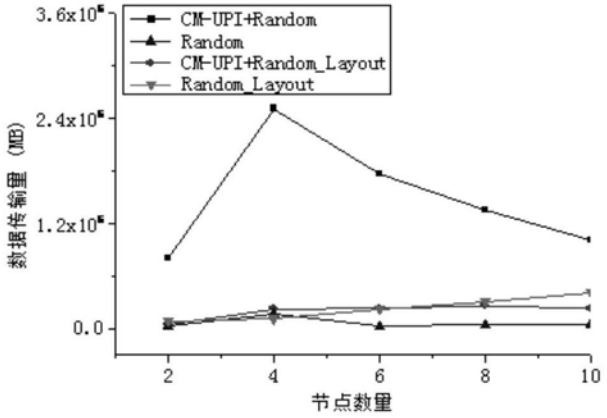

CM-UPI+Random
Random
CM-UPI+Random_Layout
Random_Layout
3.6x10⁵
2.4x10⁵
1.2x10⁵
0.0
数据传输量 (MB)
2
4
6
8
10
节点数量

图7

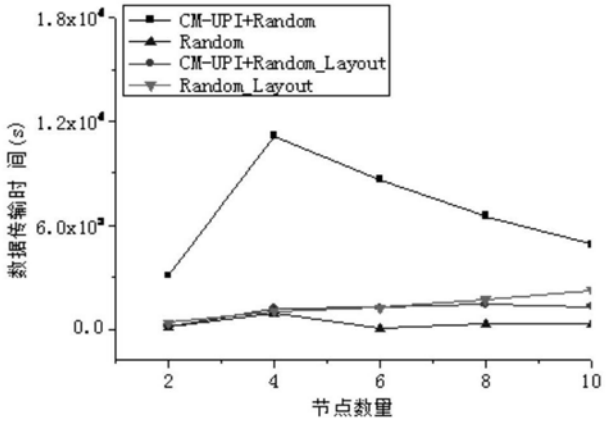

CM-UPI+Random
Random
CM-UPI+Random_Layout
Random_Layout
1.8x10⁴
1.2x10⁴
6.0x10³
0.0
数据传输时 间(s)
2
4
6
8
10
节点数量

图8

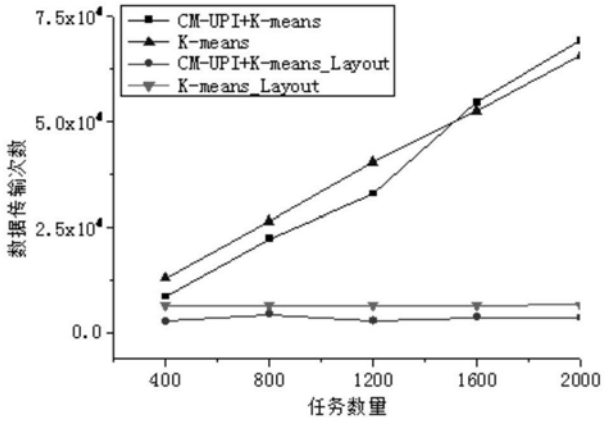

CM-UPI+K-means
K-means
CM-UPI+K-means_Layout
K-means_Layout
数据传输次数
7.5x10⁴
5.0x10⁴
2.5x10⁴
0.0
400
800
1200
1600
2000
任务数量

图9

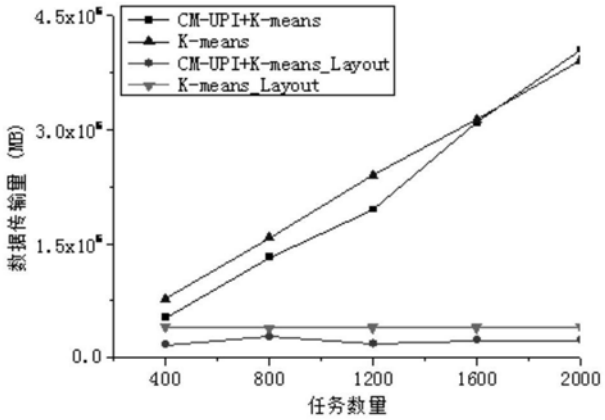

CM-UPI+K-means
K-means
CM-UPI+K-means_Layout
K-means_Layout
数据传输量 (MB)
4.5x10⁵
3.0x10⁵
1.5x10⁵
0.0
400
800
1200
1600
2000
任务数量

图10

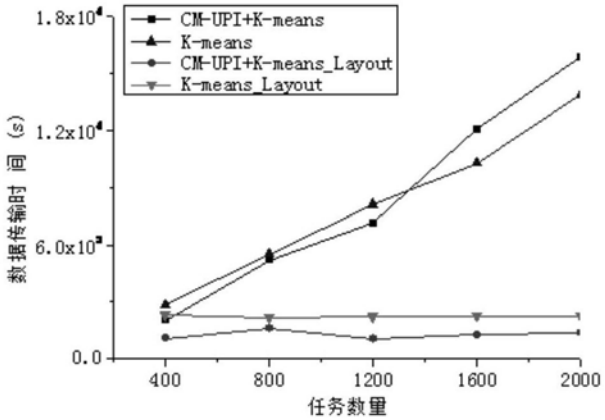

CM-UPI+K-means
K-means
CM-UPI+K-means_Layout
K-means_Layout
数据传输时 间 (s)
1.8x10⁴
1.2x10⁴
6.0x10³
0.0
400
800
1200
1600
2000
任务数量

图11

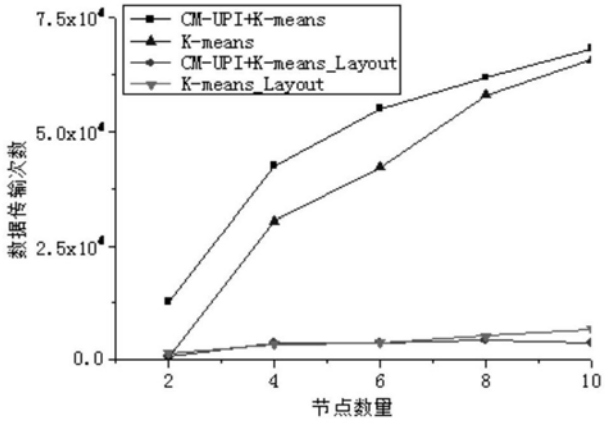

CM-UPI+K-means
K-means
CM-UPI+K-means_Layout
K-means_Layout
数据传输次数
节点数量

图12

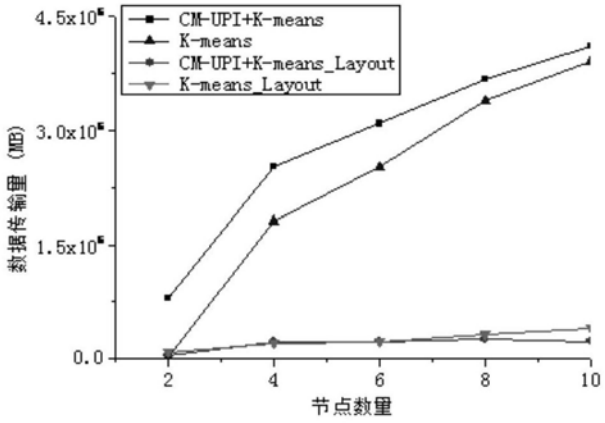

CM-UPI+K-means
K-means
CM-UPI+K-means_Layout
K-means_Layout
数据传输量（MB）
节点数量

图13

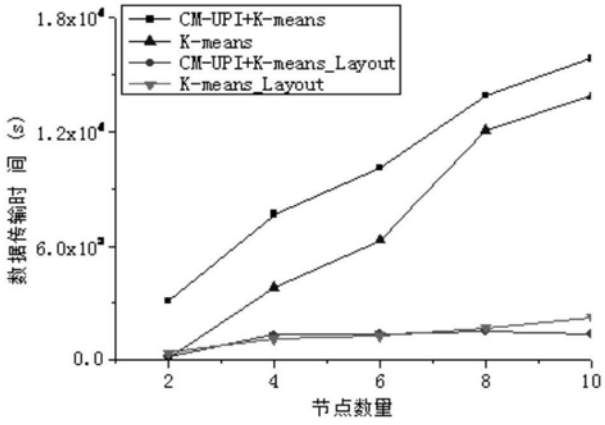

CM-UPI+K-means
K-means
CM-UPI+K-means_Layout
K-means_Layout
数据传输时间（s）
节点数量

图14