软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2016,27(7):1861−1875 [doi: 10.13328/j.cnki.jos.004879]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

混合云中面向数据中心的工作流数据布局方法*

李学俊[1,2], 吴 洋[2], 刘 晓[3], 程慧敏[2], 朱二周[2], 杨 耘[2,4]

[1](计算智能与信号处理教育部重点实验室(安徽大学),安徽 合肥 230039)

[2](安徽大学 计算机科学与技术学院,安徽 合肥 230601)

[3](华东师范大学 软件学院,上海 200062)

[4](School of Information Technology, Swinburne University of Technology, Melbourne 3122, Australia)

通讯作者: 刘晓, E-mail: xliu@sei.ecnu.edu.cn, http://www.ecnu.edu.cn

摘 要: 科学工作流是一种复杂的数据密集型应用程序.如何在混合云环境中对数据进行有效布局,是科学工作流所面临的重要问题,尤其是混合云的安全性要求给科学云工作流数据布局研究带来了新的挑战.传统数据布局方法大多采用基于负载均衡的划分模型布局数据集,该方法可以获得很好的负载平衡布局,然而传输时间并非最优.针对传统数据布局方法的不足,并结合混合云中数据布局的特点,首先设计一种基于数据依赖破坏度的矩阵划分模型,生成对数据依赖度破坏最小的划分;然后提出一种面向数据中心的数据布局方法,该方法依据划分模型将依赖度高的数据集尽量放在同一数据中心,从而减少数据集跨数据中心的传输时间.实验结果表明,该方法能够有效地缩短科学工作流运行时跨数据中心的数据传输时间.

关键词: 科学工作流;云计算;混合云;数据布局;传输时间

中图法分类号: TP316

中文引用格式: 李学俊,吴洋,刘晓,程慧敏,朱二周,杨耘.混合云中面向数据中心的工作流数据布局方法.软件学报,2016,27(7):1861−1875. http://www.jos.org.cn/1000-9825/4879.htm

英文引用格式: Li XJ, Wu Y, Liu X, Cheng HM, Zhu EZ, Yang Y. Datacenter-Oriented data placement strategy of workflows in hybrid cloud. Ruan Jian Xue Bao/Journal of Software, 2016,27(7):1861−1875 (in Chinese). http://www.jos.org.cn/1000-9825/4879.htm

Datacenter-Oriented Data Placement Strategy of Workflows in Hybrid Cloud

LI Xue-Jun[1,2], WU Yang[2], LIU Xiao[3], CHENG Hui-Min[2], ZHU Er-Zhou[2], YANG Yun[2,4]

[1](Key Laboratory of Intelligent Computing & Signal Processing of Ministry of Education (Anhui University), Hefei 230039, China)

[2](School of Computer Science and Technology, Anhui University, Hefei 230601, China)

[3](Software Engineering Institute, East China Normal University, Shanghai 200062, China)

[4](School of Information Technology, Swinburne University of Technology, Melbourne 3122, Australia)

Abstract: Scientific workflow is a complicated data intensive application. How to achieve an effective data placement schema in hybrid cloud environment has become a crucial issue nowadays, especially with the new challenges brought by the security issues. Traditional data placement strategies usually adopt load balancing-based partition model to allocate datasets. Although these data placement schemas can have good performance in load balancing, their data transfer time may not be optimal. In contrast to traditional strategies, this paper focuses on the hybrid cloud environment and proposes a data dependency destruction-based partition model to achieve the minimal data dependency destruction partition. In addition, it presents a novel datacenter-oriented data placement strategy. This strategy allocates high

* 基金项目: 国家自然科学基金(61300042, 61300169)
Foundation item: National Natural Science Foundation of China (61300042, 61300169)
收稿时间: 2014-09-13; 修改时间: 2015-06-11; 采用时间: 2015-08-12; jos 在线出版时间: 2015-12-23
CNKI 网络优先出版: 2015-12-23 10:43:25, http://www.cnki.net/kcms/detail/11.2560.TP.20151223.1043.001.html

dependency datasets to one datacenter according to the new partition model and thus significantly reduces data transfer time between datacenters. Experimental results show that the proposed strategy can effectively reduce data transfer time during workflow's execution.
Key words: scientific workflow; cloud computing; hybrid cloud; data placement; transfer time

混合云是一种公有云和私有云的混合计算环境:公有云是一种基于收费模型的提供基础设施、平台和软件等服务的计算环境[1,2],私有云是一个公司建立和管理的内部数据中心或其他 IT 组织.混合云比公有云具有更高的安全性和灵活性[3],可以根据工作负载提供高伸缩性、低成本、高安全性的服务[4].当工作负载剧烈波动时,可以通过使用公有云资源来维持服务水平,扩大私有云的能力[5].

科学工作流是基于数据驱动的应用程序,也是一种由图的节点分析操作和边缘数据流操作组成的依赖关系[6].科学工作流系统已经广泛应用于科学研究中的天文[7]、高能物理[8]等领域.例如,澳大利亚斯文本科技大学通过帕克斯无线望远镜观察数据进行的脉冲星搜索实验[9].脉冲星搜索是一个典型的数据密集型科学应用,包含复杂又费时的工作流活动,且需要处理 TB 甚至 PB 级数据[9].正因为这些科学研究所具有的数据量大、计算复杂等特点,使得科学工作流系统对其部署环境的数据存储和计算能力要求非常高.云计算以其特有的优势为科学工作流提供了一套高伸缩性、低成本的解决方案.公有云通常由 Google,Amazon 等公有云服务商提供,拥有近无限的存储和计算服务.当前,已有一些部署在云环境上的科学工作流程序,如 Nimbus[10]和 Cumulus[11]等.Deelman 等人[12]提出,云计算能够为科学工作流提供一个十分经济的部署方案[13].然而,公有云并不能为科学研究中的隐私数据提供有效的安全保障.针对公有云的不足,混合云为科学工作流提供了一种高伸缩性、低成本且高安全性的解决方案.目前,混合云通常是一种公有云和若干私有云的混合结构.

随着大数据时代的来临,混合云环境下的科学工作流数据布局已成为科学工作流研究的热点.科学工作流部署到混合云上,需要公有云和私有云相互协作,这也给数据布局带来新的挑战.混合云环境下的科学工作流数据存储,在公有云和私有云数据中心所组成的多个数据中心中具有以下特点:

- 公有云数据中心能够为用户提供无限容量的低成本存储空间,但却缺少对隐私数据的安全保障[1,2];
- 私有云数据中心由云服务商提供或由公司自己建立和管理,其建立费用高、存储容量有限、扩展费用高,所以私有云数据中心一般用来存储有限的隐私数据和热点数据;
- 隐私数据存储在固定的私有云数据中心,以保证数据的安全性[14].

传统的科学工作流数据布局方法主要包括智能方法[15,16]和聚类方法[8,14,17,18].本文针对传统数据布局方法中的聚类方法进行研究.智能方法的时间复杂度高,得到的布局结果不唯一,实用性不强.聚类方法是在基于负载均衡的基础上对数据进行布局,若在混合云环境下使用该方法,非隐私数据会被均匀地布局到所有的数据中心.该方法并没有考虑到混合云中公/私有云数据中心之间的差异.在混合云中,私有云数据中心容量有限,所有的隐私数据和部分非隐私数据布局在特定的私有云数据中心;公有云数据中心拥有近无限的存储空间,部分非隐私数据布局在公有云数据中心.因此在混合云中,基于负载均衡的布局方法并不能有效地降低数据之间的传输时间.针对传统布局方法的不足,本文首先提出一种基于数据依赖破坏度的矩阵划分模型;然后,结合混合云环境下科学工作流数据布局的特点,提出一种面向数据中心的 K 均值数据布局方法(datacenter oriented K-means,简称 DCO-K-means),包括构建阶段的静态布局算法和运行阶段的动态布局算法.传统数据布局方法首先将数据集划分成 K 个数据块,再对数据块的存放位置进行布局[8,14].本文的 DCO-K-means 方法首先以数据中心为聚类中心对数据集进行聚类划分,再根据存储在各个数据中心中非隐私数据集和隐私数据集之间的依赖关系,利用基于数据依赖破坏度的矩阵划分模型将数据集划分成数据块,最终确定数据的布局方案.与传统的数据布局方法相比,DCO-K-means 方法能够更加直接、有效地缩短混合云计算环境下科学工作流执行时数据的跨数据中心传输时间.

本文第 1 节介绍相关工作.第 2 节给出相关定义及问题分析.第 3 节设计一种基于数据依赖破坏度的矩阵划分模型.第 4 节提出一种面向数据中心的数据布局方法 DCO-K-means.第 5 节给出 DCO-K-means 方法和传统方法之间的对比实验分析与总结.第 6 节是结论与未来工作展望.

1 相关工作

传统公有云的数据存储基础设施对用户是透明的,且数据在系统内部数据中心移动不计传输时间[8,14,18].混合云环境下的科学工作流主要应用于跨学科的科学研究,这就意味着存在多个应用程序分布在不同的数据中心上执行,并且不同学科的数据在混合云环境下存放的位置也不同.所以科学工作流在混合云环境中执行时会产生大量的数据移动、大量的时间消耗以及高昂的传输费用.因此,对科学工作流数据进行合理的布局尤为重要.

文献[15,16]利用智能方法对数据进行布局.文献[15]提出了一种包含 3 个阶段的降低数据集传输时间的数据布局方法,该方法基于遗传算法,通过选择、交叉和变异来获得能够缩短跨数据中心数据传输时间的数据布局方案.文献[16]提出了一种基于粒子群算法的启发式方法来降低工作流执行时的传输费用和计算费用,该方法在数据中心的负载均衡、算法收敛方面有很好的效果.但是上述的随机化搜索方法时间复杂度高,得到的数据布局方案不唯一,且易陷入局部最优,因此实用性不高.

目前也有使用聚类方法进行数据布局的研究.Yuan[8]提出了一种减少数据集移动次数的数据布局策略,该策略利用 BEA(bond energy algorithm)算法对数据集之间的依赖矩阵进行聚类变换,并根据聚类变换得到的聚类矩阵进行 K 均值聚类划分,最终得到数据布局方案.张鹏等人[14]提出了适用于混合云环境的非重叠数据布局方法 CCDP4WF,该方法在使用 BEA 算法的同时考虑到隐私数据,在聚类变换后运用非重叠划分算法对数据依赖矩阵进行划分,并且在工作流运行阶段动态地调整数据布局,在保证业务流程完整性的同时减少数据传输.Deng[18]利用数据集和任务之间的依赖关系,在 Yuan 提出的策略的基础上提出了一种高效的数据集和任务的协同调度策略,将数据集以负载均衡的方式布置,同时将关系密切的任务和数据集聚集在一起,从而有效地提高了调度的效果并减少了数据集传输.

上述文献的数据划分模型均在负载均衡的基础上对矩阵进行划分.基于负载均衡的矩阵划分模型拥有很好的负载均衡效果,通过使用 BEA 算法将数据集聚类变换成 K 个数据块,这些块之间依然存在若干依赖关系较高的数据,若仅考虑负载均衡,会导致同一类任务相关的数据存储在不同的数据中心上.科学工作流在执行过程中,这些相关联的数据会被传输到任务执行时的数据中心,产生高额的跨数据中心时间开销.因此在混合云环境中,本文在负载均衡的基础上充分考虑不同划分块数据之间的数据依赖关系,将数据块之间与隐私数据依赖度高的非隐私数据集筛选出来,放在相同的私有云数据中心,从而有效地缩短了数据的传输时间.

2 相关定义和问题分析

本节首先对混合云环境下数据密集型工作流数据布局问题的相关概念进行定义和建模,然后分析具有代表性的基于负载均衡的矩阵划分模型、过程及存在的问题.

2.1 相关定义

科学工作流的数据集按照来源可分为初始数据集和生成数据集:初始数据集作为工作流任务的输入;生成数据集作为工作流任务的输出,然后该生成数据集又作为另一个工作流任务的输入,因此工作流数据集之间存在着非常重要的依赖关系[14].在描述工作流时,可以将工作流看成是一个有向无环图[8],工作流的任务之间存在着先后关系,当前任务必须在执行完毕后才能执行下一个任务.任务和数据之间的关系为多对多的关系,即:一个任务执行时需要多个数据集,一个数据集会被多个任务使用.

定义 1. 科学工作流.

科学工作流定义为三元组 $G=\langle T,C,DS\rangle$.其中,

- $T=\{t_1,t_2,\ldots,t_n\}$表示工作流 G 的所有任务;
- C 表示任务之间控制流的邻接矩阵,如果 $C_{ij}=1$,表示任务 t_j 必须在任务 t_i 执行完毕后才能执行;
- $DS=\{d_1,d_2,\ldots,d_n\}$表示 G 中所有数据集的集合.

定义 2. 任务.

对科学工作流中的任务,本文只关注其对应的输入数据集和生成数据集.G 中任务 t_i 为$\langle ID,OD\rangle$,i=1,2,…,$|T|$.其中,ID 表示任务 t_i 的输入数据集所组成的集合,OD 表示 t_i 的输出数据集集合.

定义 3. 数据集.

数据集 d_i 定义为$\langle ds,ptl,gt,pc\rangle$,$i$=1,2,…,$|DS|$.其中,$ds$ 表示数据集 d_i 的大小,ptl 表示数据集的存储位置,gt 表示生成数据集 d_i 的任务,pc 表示数据集 d_i 的布局位置.

ptl 和 gt 的取值如下:

$$ptl=\begin{cases}0, & d_i\in DS_{pub}\\ pricloud(d_i), & d_i\in DS_{prc}\end{cases},gt=\begin{cases}0, & d_i\in DS_{ini}\\ Task(d_i), & d_i\in DS_{gen}\end{cases}.$$

DS_{pub} 为 DS 中所有非隐私数据集集合,DS_{prc} 为 DS 中所有隐私数据集组成的集合.对$\forall d_i\in DS_{prc}$,记 d_i 指定存放的私有云数据中心为 $pricloud(d_i)$.记 DS_{ini} 为工作流 G 中所有初始数据集,DS_{gen} 为 G 在执行过程中产生的生成数据集.对$\forall d_i\in DS_{gen}$,$Task(d_i)$表示生成数据集 d_i 的任务.

定义 4. 数据中心.

混合云环境中的数据中心集合定义为 $DC=\{DC_{pri},DC_{pub}\}$.该环境下私有云数据中心集合和公有云数据中心集合分别表示为

$$DC_{pri}=\bigcup_{i=1,2,\ldots,|DC_{pri}|}dc_i,DC_{pub}=\bigcup_{j=1,2,\ldots,|DC_{pub}|}dc_j,$$

其中,数据中心可描述成 $dc_j=\{dsPrcCount,dsPubCount,size,availSize,type\}$,$j$=1,2,…,$|DC|$.$type$=1 时,$dsPrcCount$ 和 $dsPubCount$ 分别表示私有云数据中心隐私和非隐私数据集个数;$type$=0 时,$dsPubCount$ 表示公有云数据中心数据集个数,$dsPrcCount$=0.$size$ 表示数据中心的存储容量,$availSize$ 表示数据中心当前的可用存储容量.

定义 5. 数据布局方案.

工作流 $G=\langle T,C,DS\rangle$的一个数据布局方案是从集合 DS 到集合 DC 的一个映射,对$\forall d_i\in DS$,$\exists!dc_j\in DC$ 与之对应.记 $G=\langle T,C,DS\rangle$的一个数据布局方案为 PM,则有 $PM=\bigcup_{i=1,2,\ldots,|DS|}\{d_i\rightarrow dc_j\}$, $dc_j\in DC$.对$\forall d_i\in DS$,记 $PM(d_i)$为在给定数据布局方案 PM 中数据集 d_i 所对应布局位置的数据中心编号.

定义 6. 数据集之间的依赖度.

$$dependency_{ij}=Count(d_i.T\cap d_j.T),$$

其中,$d_i.T$ 和 $d_j.T$ 分别表示以数据集 d_i 和 d_j 作为输入数据集的任务集合,$Count(d_i.T\cap d_j.T)$[8,15]表示同时以数据集 d_i 和数据集 d_j 作为输入数据集的任务的个数.

定义 7. 数据依赖矩阵.

数据依赖矩阵 DM(dependency matrix)表示所有数据集间的依赖度,它是一个 $n\times n$ 的对称矩阵,其中,n 是数据集的个数.DM_{ij} 表示 d_i 和 d_j 之间的数据依赖度[8],表示为

$$DM_{ij}=dependency_{ij}.$$

类似地,

- PDM(privacy dependency matrix)表示隐私数据集之间的数据依赖矩阵;
- NPDM(non-privacy dependency matrix)表示非隐私数据集之间的数据依赖矩阵.

PDM 和 NPDM 可以从 DM 中获得.

定义 8(聚类矩阵).

聚类矩阵 CM(cluster matrix)是通过 BEA 算法转换 DM 生成的矩阵,该算法交换 DM 的行和列,使得依赖度高的数据集聚集在一起,也就是使 GM 取得最大值的排列矩阵.依赖矩阵 DM 的全局变量 GM(global measure)[8,17,19]的表示为

$$GM=\sum_{i=1}^{n}\sum_{j=1}^{n}DM_{ij}(DM_{i,j-1}+DM_{i,j+1}).$$

类似地,PCM(privacy clustered matrix)表示隐私数据集的聚类依赖矩阵,NPCM(non-privacy clustered matrix)表示非隐私数据集的聚类依赖矩阵,PCM 和 NPCM 从 CM 中获得.

2.2 问题分析

2.2.1 传统矩阵划分模型

传统矩阵划分模型是基于负载均衡的划分模型[8,14,17],该模型能够均匀地将数据分布在数据库中.Yuan[8]首次将其应用于云环境下科学工作流的数据布局领域,该模型在减少数据移动次数的同时,还能够很好地均衡云数据中心负载.

定义 9. 基于负载均衡的矩阵划分模型.

$$\max z=Sum(CU)\times Sum(CL)-Sum(CI)^2,$$

其中,$Sum(CU)$,$Sum(CL)$和 $Sum(CI)$分别表示子矩阵 CU,CL 和 CI 中所有元素之和.该模型通过在矩阵主对角线上选取 z 值最大的点进行划分,将矩阵分割成图 1 所示的 CU,CL 和 CI 子块.文献[8]指出:当 $Sum(CU)$和 $Sum(CL)$越接近时,z 值越大,数据中心的负载越均衡.但得到的布局方案的传输时间并非最优,具体分析请参见第 2.4 节.

图 1 为一个矩阵划分实例,其中,$d_1,d_2,\ldots,d_n$表示工作流应用中的数据集,矩阵中的非负整数表示每个数据集被不同任务所使用次数之和.该矩阵被划分成 4 个子矩阵,其中,左上块用 CL 表示,左下/右上块用 CI 表示,右下块用 CU 表示.

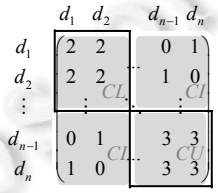


Fig.1 Matrix partition

图 1 矩阵划分

2.2.2 传统划分过程

矩阵划分过程通常分为静态数据布局和动态数据布局.静态数据布局分为 3 步:首先构造数据依赖矩阵 DM,由 DM 得到 PDM 和 NPDM;然后,将非隐私数据依赖矩阵 NPDM 转换为聚类矩阵 NPCM;最后,将矩阵划分为若干数据块.动态布局和静态布局类似,工作流在运行过程中,将产生的数据动态更新到静态数据布局方案中,最后生成动态数据布局方案.

图 2 是一个简单的工作流示例,它由 5 个任务 t_1~t_5、6 个初始数据集 d_1~d_6 和 5 个生成数据集 d_7~d_{11} 组成.其中,d_1~d_4,d_7~d_{11} 为非隐私数据集,d_5 和 d_6 为隐私数据集.非隐私数据存储在公有云数据中心 dc_1 和私有云数据中心 dc_2,dc_3 上,隐私数据集 d_5 存放在 dc_2,d_6 存放在 dc_2.静态数据布局只对 d_1~d_6 进行布局,矩阵划分过程如后文图 3~图 6 所示.

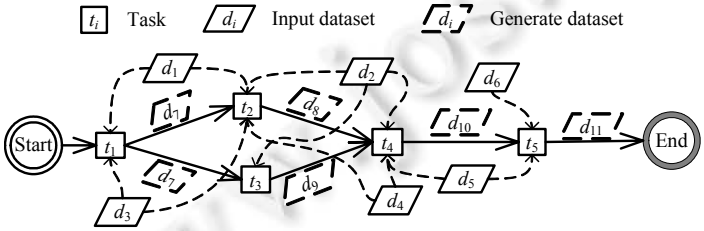


Fig.2 Example of workflow

图 2 工作流示例

(1) DM 构建

对于图 2 给出的工作流示例,确定 d_1~d_6 和任务之间的关联关系构建 DM,如图 3 所示.

$$\begin{array}{l} d_1.T=\{t_1,t_2\} \\ d_2.T=\{t_2,t_3,t_4\} \\ d_3.T=\{t_1,t_2\} \\ d_4.T=\{t_2,t_4\} \\ d_5.T=\{t_4,t_5\} \\ d_6.T=\{t_5\} \end{array} \Rightarrow \begin{array}{c} \\ d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \\ d_6 \end{array} \begin{array}{c} \begin{array}{cccccc} d_1 & d_2 & d_3 & d_4 & d_5 & d_6 \end{array} \\ \begin{pmatrix} 2 & 1 & 2 & 1 & 0 & 0 \\ 1 & 3 & 1 & 2 & 1 & 0 \\ 2 & 1 & 2 & 1 & 0 & 0 \\ 1 & 2 & 1 & 2 & 1 & 0 \\ 0 & 1 & 0 & 1 & 2 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix} \end{array}$$

Fig.3 Build DM

图 3 DM 构建

(2) PDM 和 NPDM 构建

根据 DM 中隐私数据和非隐私数据分别构建 PDM 和 NPDM,如图 4 所示.

$$\begin{array}{c} \\ d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \\ d_6 \end{array} \begin{array}{c} \begin{array}{cccccc} d_1 & d_2 & d_3 & d_4 & d_5 & d_6 \end{array} \\ \begin{pmatrix} 2 & 1 & 2 & 1 & 0 & 0 \\ 1 & 3 & 1 & 2 & 1 & 0 \\ 2 & 1 & 2 & 1 & 0 & 0 \\ 1 & 2 & 1 & 2 & 1 & 0 \\ 0 & 1 & 0 & 1 & 2 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix} \end{array} \Rightarrow \begin{array}{c} \\ d_5 \\ d_6 \end{array} \begin{array}{c} \begin{array}{cc} d_5 & d_6 \end{array} \\ \begin{pmatrix} 2 & 1 \\ 1 & 1 \end{pmatrix} \end{array} + \begin{array}{c} \\ d_1 \\ d_2 \\ d_3 \\ d_4 \end{array} \begin{array}{c} \begin{array}{cccc} d_1 & d_2 & d_3 & d_4 \end{array} \\ \begin{pmatrix} 2 & 1 & 2 & 1 \\ 1 & 3 & 1 & 2 \\ 2 & 1 & 2 & 1 \\ 1 & 2 & 1 & 2 \end{pmatrix} \end{array}$$

PDM NPDM

Fig.4 Build PDM and NPDM

图 4 PDM 和 NPDM 构建

(3) NPDM 转换成 NPCM

NPDM 利用 BEA 算法变换为聚类矩阵 NPCM,如图 5 所示.

(4) NPCM 划分

利用定义 9 对 NPCM 进行划分,从矩阵的对角线上选择使 z 值最大的点进行划分,最终得到分割成与数据中心个数相同的划分块,如图 6 所示.数据集划分成$\{d_1,d_3\}$,$\{d_2\}$和$\{d_4\}$这 3 个部分.

$$\begin{array}{c} \\ d_1 \\ d_2 \\ d_3 \\ d_4 \end{array} \begin{array}{c} \begin{array}{cccc} d_1 & d_2 & d_3 & d_4 \end{array} \\ \begin{pmatrix} 2 & 1 & 2 & 1 \\ 1 & 3 & 1 & 2 \\ 2 & 1 & 2 & 1 \\ 1 & 2 & 1 & 2 \end{pmatrix} \end{array} \Rightarrow \begin{array}{c} \\ d_1 \\ d_2 \\ d_3 \\ d_4 \end{array} \begin{array}{c} \begin{array}{cccc} d_1 & d_2 & d_3 & d_4 \end{array} \\ \begin{pmatrix} 2 & 2 & 1 & 1 \\ 2 & 2 & 1 & 1 \\ 1 & 1 & 3 & 2 \\ 1 & 1 & 2 & 2 \end{pmatrix} \end{array}$$

Fig.5 Transform NPDM into NPCM

图 5 NPDM 转换成 NPCM

$$\begin{array}{c} \\ d_1 \\ d_2 \\ d_3 \\ d_4 \end{array} \begin{array}{c} \begin{array}{cccc} d_1 & d_2 & d_3 & d_4 \end{array} \\ \begin{pmatrix} 2 & 2 & 1 & 1 \\ 2 & 2 & 1 & 1 \\ 1 & 1 & 3 & 2 \\ 1 & 1 & 2 & 2 \end{pmatrix} \end{array}$$

Fig.6 Partition NPCM

图 6 NPCM 划分

2.2.3 数据布局分析

混合云环境下,科学工作流在执行任务时需要将该任务的所有输入数据集传输到任务的执行位置,相应的输入数据集在任务执行前没有存放到任务的执行位置时,则需要对数据进行跨数据中心传输.

由于科学工作流中向一个数据中心传输数据的传输时间远大于向该数据中心调度任务的时间,所以科学工作流的任务调度策略为:将任务调度到该任务执行所需跨数据中心传输时间最小的数据中心,从而有效地缩短数据的传输时间[8,14,18].

图 2 中的工作流示例采用传统矩阵划分模型划分结果如图 7 所示,最优的数据布局方案如图 8 所示.在混合云计算环境中,私有云数据中心存在容量限制,为了展示和分析数据布局方案,图 7 和图 8 中的私有云数据中

心没有考虑容量限制因素.

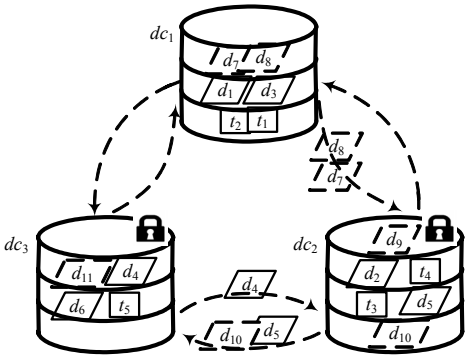


Fig.7 Traditional data placement layout

图 7 传统划分模型生成的数据布局

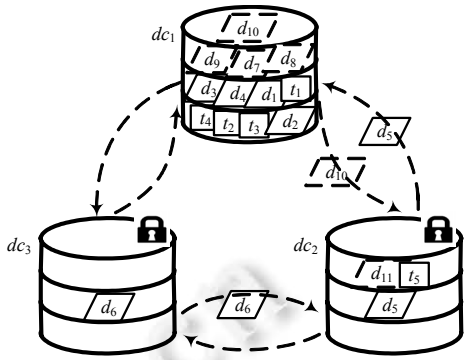


Fig.8 Optimal data placement layout

图 8 最优数据布局

图 7 是采用传统划分模型生成的数据布局方案.分为两个阶段:静态数据布局和动态数据布局.静态数据布局为$\{d_1,d_3\}$存储在 dc_1,$\{d_2,d_5\}$存储在 dc_2,$\{d_4,d_6\}$存储在 dc_3;$\{t_1,t_2\}$调度到 dc_1,$\{t_3,t_4\}$调度到 dc_2,$\{t_5\}$调度到 dc_3;生成数据集存储在生成它的任务所在的数据中心.动态数据布局过程中,d_4 从 dc_3 传输到 dc_2 一次,d_5 从 dc_2 传输到 dc_3 一次,d_7 从 dc_1 传输到 dc_2 一次,d_8 从 dc_1 传输到 dc_2 一次,d_{10} 从 dc_2 传输到 dc_3 一次,传输次数总计为 5.

下面给出最优的数据布局方案,也分为静态和动态数据布局,如图 8 所示.静态数据布局将 d_1~d_6 划分为$\{d_1,d_2,d_3,d_4\}$,$\{d_5\}$和$\{d_6\}$这 3 个部分(如第 4 节中的图 9、图 10 所示),其中,$\{d_1,d_2,d_3,d_4\}$存储在 dc_1,$\{d_5\}$存储在 dc_2,$\{d_6\}$存储在 dc_3;$\{t_1,t_2,t_3,t_4\}$调度到 dc_1,$\{t_5\}$调度到 dc_2;生成数据集存储在生成它的任务所在的数据中心.动态数据布局过程中,d_5 从 dc_2 传输到 dc_1 一次,d_6 从 dc_3 传输到 dc_2 一次,生成数据 d_{10} 从 dc_1 传输到 dc_2 一次,传输次数总计为 3.显然,最优数据布局的传输时间比传统划分模型要少.因此,该划分模型不是最优的.

传统划分模型是基于数据负载均衡对数据进行划分,划分的数据均衡地布局到所有的数据中心,数据中心的负载均衡最优.但是该模型使得依赖度较高的数据可能没有存放到同一个数据中心,从而增加了工作流执行阶段数据集跨数据中心的传输时间.公有云拥有海量的存储和计算资源,如果不能有效地利用这些资源,势必会增加工作流的运行和维护成本.因此,本文针对传统数据布局模型的不足,设计出一种新的基于数据依赖破坏度的矩阵划分模型.该模型在综合考虑数据类型的基础上,尽量减少划分对数据依赖造成的破坏,使数据依赖度高的数据尽可能地划分到同一个数据中心上.这样既能够降低私有云数据中心出现过载问题,又能够提高公有云数据中心的利用率.

3 基于数据依赖破坏度的矩阵划分模型

本节针对传统基于数据负载均衡的矩阵划分模型的不足,提出一种新的基于数据依赖破坏度的矩阵划分模型.下面首先给出静态布局阶段初始数据集相关的定义和划分过程,然后给出动态布局阶段生成数据集相关的定义.

定义 10. 数据依赖破坏度.

在工作流 $G=\langle T,C,DS\rangle$的一个数据布局方案 PM 中,若$\exists d_i,d_j\in DS$,使得 $dependency_{ij}\neq 0$,并且 $PM(d_i)\neq PM(d_j)$,则称 PM 对 G 的数据依赖造成破坏,用 DD(dependency destruction)表示数据依赖破坏度为

$$DD_{ij}=\begin{cases}0, & PM(d_i)=PM(d_j)\\ dependency_{ij}, & PM(d_i)\neq PM(d_j)\end{cases}.$$

定义 11. 基于数据依赖破坏度的矩阵划分模型.

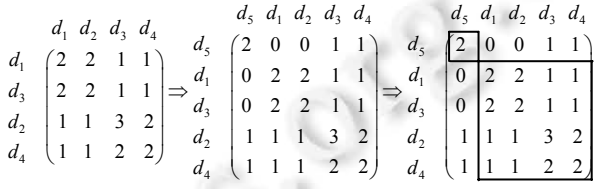
该模型表示对数据集依赖破坏度最小的划分,是 CI 中所有元素之和,表示为

$$\min\ z=\sum\sum DD_{ij},$$

其中,i=1,…,$row(CI)$,j=1,…,$column(CI)$,其中,$row(CI)$和 $column(CI)$分别表示 CI 的行数和列数.

与传统划分模型不同,该模型表示工作流在执行过程中同一个任务同时使用 CU 和 CL 中数据集所产生的跨数据传输次数,能够尽量减少数据布局对数据依赖的破坏.与第 2.4 节中的传统划分模型相比,该模型数据布局方案能够有效地缩短数据传输时间.

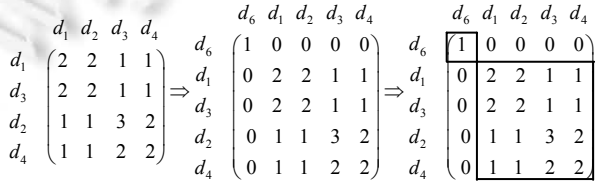
基于该划分模型的矩阵划分过程,与传统的划分过程只是在第 4 步不同.传统的划分模型直接将 NPCM 划分到所有的数据中心.但是该划分过程首先将 PDM 中所有隐私数据集 d_i 按照私有云数据中心的不同依次添加到 NPCM,每添加一个私有云数据中心中的所有 d_i 到 NPCM 便使用定义 11 对矩阵进行垂直划分;然后,将与 d_i 依赖度高的非隐私数据集布局到 d_i 所对应的数据中心;PDM 中的隐私数据全部添加到 NPCM 后,再将 NPCM 中剩余的非隐私数据集全部布局到公有云数据中心.图 2 所示的工作流示例中依据隐私数据集 d_5 和 d_6,静态阶段划分非隐私数据集 d_1~d_4 的过程,分别如图 9 和图 10 所示.



(a) NPCM (b) CM (c) Partitioned CM

Fig.9 Process of NPCM partition based on d_5

图 9 依据 d_5 划分 NPCM 的过程



(a) *NPCM* (b) CM (c) Partitioned CM

Fig.10 Process of NPCM partition based on d_6

图 10 依据 d_6 划分 NPCM 的过程

基于数据依赖破坏度的划分过程将图 2 所示工作流示例中的数据集划分成{d_1,d_2,d_3,d_4},{d_5}和{d_6}这 3 个部分,并进行布局.

上述的模型和划分过程仅涉及到工作流静态布局的初始数据集,动态布局中生成数据集相关的模型如定义 12 和定义 13 所示.

定义 12. 生成数据集和私有云数据中心之间的依赖度.

假设 $d_i\in DS_{gen}$,$dc_j\in DC_{pri}$,则生成数据集 d_i 和私有云数据中心 dc_j 之间的依赖度是 d_i 和 dc_j 中所有数据集依赖度之和,表示为

$$dependency_{ij}^{dc_j}=\sum_{k=1}^{dc_j.dsPrcCount}{}_{d_k\in dc_j}\,dependency_{ik},$$

其中,$dc_j.type$=1,j=1,2,…,$|DC_{pri}|$.

定义 13. 生成数据集和私有云数据中心的最大依赖度.

假设 $d_i\in DS_{gen}$,$dc_j\in DC_{pri}$,则生成数据集 d_i 布局到依赖度最高的私有云数据中心 dc_k,其中,k 取值为

$$k=\max_{j}\{dependency_{ij}^{dc_j}\}.$$

4 面向数据中心的数据布局方法 DCO-*K*-means

面向数据中心的数据布局方法 DCO-*K*-means 包括静态数据布局算法和动态数据布局算法.静态数据布局算法在工作流的构建阶段通过矩阵划分模型对初始数据集进行布局,生成构建阶段的数据布局方案;动态数据布局算法在工作流运行阶段利用矩阵划分模型对生成数据集进行布局,得到传输时间最优的布局方案.

4.1 静态数据布局算法

该算法是一种新的基于数据中心的聚类算法对数据集进行聚类划分.隐私数据集存放在固定的私有云数据中心,对于非隐私数据,该方法以每个私有云数据中心中存放的隐私数据集为聚类中心,将与隐私数据依赖度高的非隐私数据布局到私有云数据中心,与隐私数据依赖度低的非隐私数据布局到公有云数据中心.

算法 1. 构建阶段 Datacenter Oriented *K*-means 算法.

输入:初始数据集 DS_{ini},数据中心 DC;

输出:数据布局方案 PM.

```
1.  for (every dc_i in DC_pri)
2.      add dc_i to DC_pri_Queue;
3.  initialize DM, NPDM;
4.  NPCM=BEA(NPDM);
5.  while (DC_pri_Queue≠NULL) {
6.      CM=NPCM
7.      for (every dc_i in DC_pri_Queue) {
8.          for (j=1; j≤||CM||; j++) {          //||CM||为聚类矩阵的行数
9.              CM′=CM;
10.             add all private dataset d_k to CM′, where d_k belongs to dc_i;
11.             transform CM′,                  //以迭代法将 CM′最左列和最右列,最上行和最下行互换
12.             partition CM′;
13.             calculate z of CM′ (定义 11);
14.         }
15.         z′=min{z};
16.     }
17.     select CM′ with min{z′};                //找出私有云数据中心中与非隐私数据依赖度最高的聚类矩阵
18.     allocate datasets in CL to dc_i;        //将 CL 中的所有数据集布局到 dc_i
19.     update dc_i to PM;
20.     delete dc_i from DC_pri_Queue;          //删除布局过的私有云数据中心
21.     NPCM=CU;
22. }
23. allocate datasets in NPCM to DC_pub;
24. update DC_pub to PM;
25. return PM;
```

步骤 1、步骤 2 建立私有云数据中心队列 DC_{pri}_Queue.步骤 3 初始化数据依赖矩阵 DM 和 NPDM.步骤 4 通过 BEA 算法对 NPDM 进行聚类,得到 NPCM.步骤 5~步骤 22 通过向 NPCM 中依次添加每个私有云数据中心的隐私数据,对新获得的 CM′进行划分,并将与私有数据集依赖度最高的非隐私数据布局到对应的私有云数

据中心.步骤23~步骤25将NPCM中尚未处理的非隐私数据集全部布局到公有云数据中心,此时布局完毕,得到数据布局方案PM.

4.2 动态数据布局算法

工作流在运行阶段会产生生成数据集.动态数据布局算法首先将所有的生成数据集更新到DM,然后将NPDM中非隐私数据集布局到与隐私数据集依赖度最高的私有云数据中心.在布局的过程中,如果数据集造成某个私有云数据中心的存储负载超过存储容量,动态数据布局算法就会对数据布局方案进行调整.

算法2. 运行阶段Datacenter Oriented *K*-means算法.

输入:生成数据集DS_{gen},数据中心DC;

输出:数据布局方案PM′.

1. add d_i to DM;
2. for (every dc_j in DC_{pri}) {
3. calculate $dependency_{ij}^{dc_j}$ (定义12);
4. select dc_k (定义13);
5. if ($d_i.ds \leqslant dc_k.availSize$) allocate d_i to dc_k;
6. else {
7. partition & allocate DM and get PM′, //调用第4.1节静态布局算法第4步~第24步
8. for (all d_i in PM′)
9. if ($PM'(d_i) \neq PM(d_i)$) update d_i to $PM'(d_i)$;
10. }
11. }
12. return PM′;

步骤1将生成数据集d_i添加到DM,步骤3、步骤4计算d_i与每个私有云数据中心中所有隐私数据集的依赖度,获得依赖度最大的私有云数据中心dc_k.步骤5~步骤10中:如果数据d_i的大小低于dc_k的剩余存储空间,则将数据集d_i布局在数据中心dc_k上;反之,则对当前的全局数据依赖矩阵DM进行聚类划分,获得新的数据布局方案PM′,PM′中的d_i被布局在另一个拥有足够容量的数据中心上.由于新获得的数据布局方案PM′和原有的PM存在差异,因此算法比较两种布局方案,并将原有数据集按照新的数据布局方案PM′进行调整.

5 实验分析

5.1 实验设置

本节将对本文提出的数据布局方法DCO-*K*-means与具有代表性的*K*-means[8]和CCDP4WF[14]传统数据布局方法进行对比实验.*K*-means方法[8]利用BEA算法,在构建阶段将初始数据集聚类变换成*K*个数据块,并分别布局到*K*个数据中心;在运行阶段,动态地将新生成的数据集存放到其依赖度最高的数据集所在的数据中心.CCDP4WF方法[14]在文献[8]的基础上,首先在构建阶段分析数据集的隐私性,将初始数据集存放到云端和客户端;在运行阶段,根据新生成数据集与初始数据集之间的依赖关系,动态地调整数据布局.本文的DCO-*K*-means方法在构建阶段,以数据中心为聚类中心对初始数据集进行聚类划分,再根据存储在各个数据中心中非隐私数据集和隐私数据集之间的依赖关系布局初始数据集;在运行阶段,同时考虑新生成数据集与已有数据集之间的依赖关系和其本身是否为隐私数据集,动态地将其布局到合适的数据中心.DCO-*K*-means方法在两个阶段均考虑数据集之间的依赖破坏度,将数据集存放到破坏度最小的数据中心,从而有效地缩短了数据的传输时间.

本文实验采用控制变量法,给出在数据集个数、私有云数据中心个数、隐私数据集百分比以及数据集大小4种参数变化下,3种布局方法的传输时间情况,具体包括:

(1) 每个数据集大小为500MB,隐私数据集百分比为10%,私有云数据中心3个,数据集个数从10递增到

90 个,每次递增 10,如后文图 11 所示;

(2) 数据集 40 个,每个数据集大小为 500MB,隐私数据集百分比为 10%,私有云数据中心从 3 个递增到 7 个,每次递增 1,如后文图 12 所示;

(3) 数据集 40 个,每个数据集大小为 500MB,私有云数据中心 3 个,隐私数据集百分比从 0 递增到 70%,每次递增 10%,如后文图 13 所示;

(4) 数据集 40 个,隐私数据集百分比为 10%,私有云数据中心 3 个,每个数据集大小从 100MB 递增到 800MB,每次递增 100MB,如后文图 14 所示.

针对每组实验中参数的不同要求,首先随机生成相应数量的数据集,再建立任务和数据集之间的依赖关系.每个工作流均由 10 个任务随机组成,所有数据集、任务存放于 1 个公有云、多个私有云数据中心所组成的混合云环境中,数据中心之间的带宽设置为 10Mbps.由于随机生成的数据集总和小于 1TB,因此假定虚拟生成 1 个存储容量为 10TB 的存储空间作为近似无限的公有云数据中心,私有云数据中心的存储容量按如下公式设置:

$$dc_j.size=|DS|\times d_i\cdot ds/(|DC|-1),$$

其中,j=1,2,…,$|DC_{pri}|$.每组实验中,工作流的参数和云平台环境相同.为确保实验结果的可靠性,每组实验运行 100 次后取平均值,作为最终的传输时间.

5.2 实验结果与分析

5.2.1 数据集个数变化对传输时间的影响

图 11 表示数据集个数增加时数据传输时间的变化趋势,其中,工作流数据集大小为 500MB,隐私数据集占所有数据集的 10%,所有数据集布局到 1 个公有云、3 个私有云数据中心组成的混合云环境中.3 种数据布局方法的传输时间均随着数据集个数的增加而上升,其中,CCDP4WF 和 *K*-means 方法的传输时间上升幅度基本相同, DCO-*K*-means 方法的增长趋势明显小于 CCDP4WF 和 *K*-means 方法.在数据集个数为 90 时,传输时间也不足 2 000s(1 815s).与 CCDP4WF(8 090s)和 *K*-means(8 680s)方法相比,大约降低了 75%,其优化效果明显.

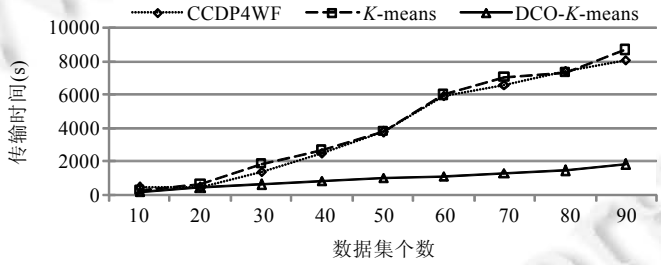


Fig.11 Comparison of transmission time with the number of datasets

图 11 不同数据集个数的传输时间

CCDP4WF 和 *K*-means 方法基于负载均衡原则将数据聚类到不同数据中心,没有充分考虑到隐私和非隐私数据之间的依赖关系,导致依赖度高的数据集被布局到不同的数据中心.因此,随着数据集个数的增加,两种方法的时间开销较高.本文的 DCO-*K*-means 方法充分考虑了数据集之间的依赖关系,在划分的过程中将依赖度高的数据集存放到同一个数据中心,以减少任务在执行时需要的跨数据中心的数据传输.所以工作流在混合云中执行时,本文方法较 CCDP4WF 和 *K*-means 方法在数据传输时间上有较好的优化效果.

5.2.2 私有云数据中心个数变化对传输时间的影响

图 12 给出公有云数据中心 1 个、私有云数据中心数量逐渐增加时数据传输时间的变化趋势.其中,工作流数据集为 40 个,每个数据集 500MB,隐私数据集占所有数据集的 10%.私有云数据中心从 3 个增加到 7 个时,3 种数据布局方法的数据传输时间均在某一水平线上下波动,其中,DCO-*K*-means 方法的稳定性最高,传输时间一直低于 1 000s,波动范围在±3.6%之间,其传输时间优化得最好,传输时间较 CCDP4WF 和 *K*-means 方法降低了约

50%.*K*-means 方法在时间上的优化次之,CCDP4WF 最差,这两种方法的传输时间都高于 2 000s.

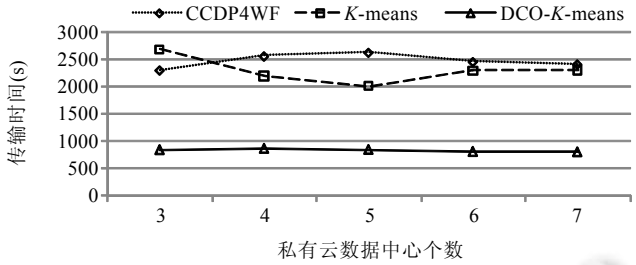


Fig.12 Comparison of transmission time with the number of private datacenters

图 12 不同私有云数据中心个数的传输时间

DCO-*K*-means 的布局方案仅与隐私数据的存放位置和数据依赖程度相关,在隐私数据集个数确定的情况下,非隐私数据集根据存储在不同数据中心之间的隐私数据集依赖关系,将依赖度高的非隐私数据集布局到对应的私有云数据中心,将依赖度低的非隐私数据集存储到公有云数据中心.因此,私有云数据中心的增加对 DCO-*K*-means 方法的数据布局不会造成大的影响,基本维持在 832s 左右的范围.然而,基于负载均衡的 CCDP4WF 和 *K*-means 方法会将所有的非隐私数据集均匀布局到逐渐增加的数据中心,破坏了数据集之间的依赖关系,导致数据传输时间不仅大于 DCO-*K*-means 方法,得到的传输时间稳定性也比 DCO-*K*-means 要差.

5.2.3 隐私数据集百分比变化对传输时间的影响

图 13 表示隐私数据集比例逐渐增加时数据传输时间的变化趋势,其中,工作流数据集为 40 个,每个数据集 500MB,所有数据集布局到 1 个公有云、3 个私有云数据中心组成的混合云环境中.隐私数据集百分比为 0 时,3 种方法的传输时间为 0s,此时,所有的数据集均布局到公有云数据中心.随着隐私数据集百分比的增加,3 种方法的传输时间逐渐增加,并且增长的幅度逐渐减小,在隐私数据集占 70%时趋于相同(4 950s).从图 13 可以看出:CCDP4WF 和 *K*-means 方法的传输时间大致一致,并在隐私数据集为 10%~40%时明显高于本文的 DCO-*K*-means 方法,平均多消耗 40%的时间.

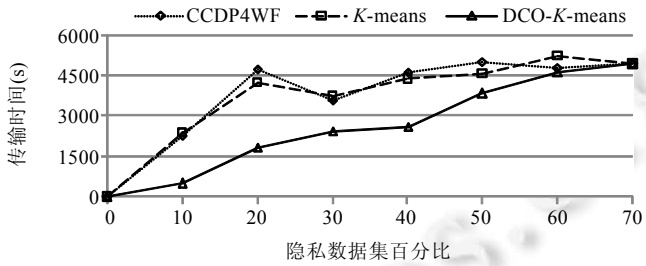


Fig.13 Comparison of transmission time with the rates of privacy datasets

图 13 不同隐私数据集百分比的传输时间

数据总量为 40 个,随着隐私数据集比例的增加,更多的隐私数据集存放在私有云数据中心,更少的非隐私数据集可供灵活布局,并且工作流数据和任务的依赖关系没有改变.所以,工作流在执行时会产生更多的数据跨数据中心传输,导致数据的传输时间显著增加.当隐私数据集比率为 0 时,3 种方法均采用相同的聚类布局算法,将所有的非隐私数据集布局到同一个公有云数据中心上,数据的传输时间也为 0.当隐私数据集比率逐渐增加时, CCDP4WF 和 *K*-means 方法以负载均衡为目标,没有充分考虑非隐私和隐私数据集之间的依赖关系,导致数据传输时间显著增加.而 DCO-*K*-means 方法在非隐私和隐私数据集依赖关系的基础上对数据集布局,因此缩短了数据的传输时间,提升了工作流的执行效率.当隐私数据比率逐渐上升到 70%左右时,3 种方法的传输时间逐

渐趋于相同,因为非隐私数据集和更多的隐私数据集存在相同的依赖度,同一个非隐私数据集有多个相同效果的布局方案,并且可供布局的非隐私数据集个数也较少.这两个主要因素导致 DCO-*K*-means 方法较 CCDP4WF 和 *K*-means 方法的优势不够明显,数据传输时间逐渐趋于相同.

5.2.4 数据集大小变化对传输时间的影响

图 14 为每个数据集大小逐渐改变时传输时间的变化趋势,其中,工作流数据集为 40 个,隐私数据集占所有数据集的 10%,所有数据集布局到 1 个公有云、3 个私有云数据中心组成的混合云环境中.本文假设该实验中所有数据集大小相同,若仅改变数据集大小而不改变工作流中任务和数据集之间的依赖关系,则不能充分反映数据集大小改变对工作流数据布局的影响状况.因此,本实验在数据集大小改变时,按照参数要求重新生成所有工作流,并在相同环境下做 100 次相同实验取平均值.可以看到:随着数据集大小的逐渐增大,3 种布局方法的数据传输时间呈近似线性增长趋势.在数据集大小为 100MB~700MB 时,CCDP4WF 方法的时间优化最差,*K*-means 次之, DCO-*K*-means 最优,分别较 CCDP4WF 和 *K*-means 方法平均节约 71%和 67%左右的时间.

CCDP4WF 和 *K*-means 方法主要从负载均衡的角度布局,破坏了原有数据中隐私和非隐私数据集之间的依赖性,导致传输时间迅速增加.而 DCO-*K*-means 方法充分考虑了隐私和非隐私数据集的依赖关系,布局时尽量减少布局对数据依赖关系的破坏,数据传输时间随着数据集的增大而小幅增加,在传输时间上,明显优于前两种布局方法.每组实验均重新生成满足相同参数约束的工作流,其中,数据集之间的依赖关系、隐私数据集的选取存在不确定性,可能导致数据集大小在 800MB 时,CCDP4WF 方法得到的布局方案突然优于 *K*-means 方法.

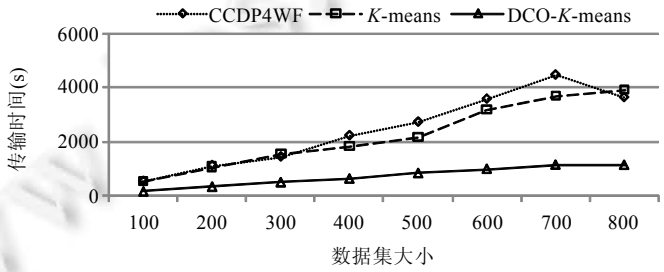


Fig.14 Comparison of transmission time with the size of datasets

图 14 不同数据集大小的传输时间比较

综合上述 4 个方面的实验结果,DCO-*K*-means 方法的布局方案均优于 *K*-means 和 CCDP4WF 方法,有效地缩短了数据传输时间.对于数据规模大、隐私数据集比率为 10%~50%的科学工作流应用,DCO-*K*-means 优化效果会更明显.

6 结论与未来工作

本文首先介绍混合云环境的特点,数据密集型应用程序部署到混合云环境带来了新的机遇与挑战.对于科学工作流的数据布局,传统的数据布局方法通常使用聚类方法将数据布局到混合云中.该方法使用基于负载均衡的矩阵划分模型将数据集均匀地布局到所有的数据中心,但是得到的数据布局方案传输时间均非最优.针对传统方法的不足,我们提出一种新的面向数据中心的数据布局优化方法 DCO-*K*-means,该方法使用基于数据依赖破坏度的矩阵划分模型,将依赖度高的数据集布局到同一个数据中心,从而减少数据传输时间.该方法包括构建阶段静态布局算法和运行阶段动态布局算法,它们分别在工作流的构建阶段和执行阶段对数据进行布局,获得传输时间最优的布局方案.最后,与传统的 CCDP4WF 和 *K*-means 方法进行对比实验分析,表明 DCO-*K*-means 方法得到的布局方案优化效果明显.

本文在实验时假定所有数据集大小相同,所有数据中心之间的带宽相同,而实际情况下,数据集的大小差异很大,并且数据中心之间的带宽是动态的,这些因素会对本文的数据布局算法造成一定的影响.对于这些局限,

我们将继续研究能够反映数据集大小的加权数据依赖矩阵、能够表示数据中心之间带宽变化的数学模型,并最后将该方法应用于具体的科学工作流领域中.

References:

[1] Armbrust M, Fox A, Griffith R, Joseph AD, Katz R, Konwinski A, Lee GH, Patterson D, Rabkin A, Stoica I, Zaharia M. Above the clouds: A Berkeley view of cloud computing. Technical Report, No.UCB/EECS-2009-28, Berkeley: Department of Electrical Engeering and Computer Sciences, University of California, 2009.

[2] Buyya R, Pandey S, Vecchiola C. Cloudbus Toolkit for Market-Oriented Cloud Computing. Berlin, Heidelberg: Springer-Verlag, 2009. 24−44. [doi: 10.1007/978-3-642-10665-1_4]

[3] Sumit G. Public vs. private vs hybrid vs community-cloud computing: A critical review. Int'l Journal of Computer Network and Information Security (IJCNIS), 2014,6(3):21−29. [doi: 10.5815/ijcnis.2014.03.03]

[4] Grewal RK, Pateriya PK. A rule-based approach for effective resource provisioning in hybrid cloud environment. In: Proc. of the New Paradigms in Internet Computing. Berlin, Heidelberg: Springer-Verlag, 2013. 41−57. [doi: 10.1007/978-3-642-35461-8_5]

[5] Fu J, Wang JC, Lu J, Chen ZH, He MQ. Research on meteorology indices forecasting framework based on hybrid cloud computing platforms. In: Proc. of the Ubiquitous Information Technologies and Applications. Netherlands: Springer-Verlag, 2013. 727−735. [doi: 10.1007/978-94-007-5857-5_78]

[6] Szabo C, Sheng QZ, Kroeger T, Zhang YH, Jian Y. Science in the cloud: Allocation and execution of data-intensive scientific workflows. Journal of Grid Computing, 2014,12(2):245−264. [doi: 10.1007/s10723-013-9282-3]

[7] Deelman E, Singh G, Sua M, Blythe J, Gil Y, Kesselman C, Mehta G, Vahi K, Berriman GB, Good J, Laity A, Jacob JC, Katz DS. Pegasus: A framework for mapping complex scientific workflows onto distributed systems. Scientific Programming, 2005,13(3): 219−237. [doi: 10.1155/2005/128026]

[8] Yuan D, Yang Y, Liu X. A data placement strategy in scientific cloud workflows. Future Generation Computer Systems, 2010, 26(8):1200−1214. [doi: 10.1016/j.future.2010.02.004]

[9] Yuan D, Yang Y, Liu X. On-Demand minimum cost benchmarking for intermediate dataset storage in scientific cloud workflow systems. Journal of Parallel & Distributed Computing, 2011,71(2):316−332. [doi: 10.1016/j.jpdc.2010.09.003]

[10] Keahey K, Figueiredo R, Freeman T, Tsugawa M. Science clouds: Early experiences in cloud computing for scientific applications. In: Proc. of the Cloud Computing and its Applications. 2008. 825−830. https://www.researchgate.net/publication/240918762_Science_Clouds_Early_Experiences_in_Cloud_Computing_for_Scientific_Applications

[11] Wang LZ, Tao J, Kunze M, Castellanos AC, Kramer D, Karl W. Scientific cloud computing: early definition and experience. In: Proc. of the 10th IEEE Int'l Conf. on High Performance Computing and Communications (HPCC 2808). 2008. 825−830. [doi: 10.1109/HPCC.2008.38]

[12] Deelman E, Singh G, Livny M, Berriman B, Good J. The cost of doing science on the cloud: The montage example. In: Proc. of the 2008 ACM/IEEE Conf. on Supercomputing. 2008. 1−12. [doi: 10.1109/SC.2008.5217932]

[13] Hoffa C, Mehta G, Freeman T, Deelman E, Keahey K, Berriman B, Good J. On the use of cloud computing for scientific workflows. In: Proc. of the IEEE 4th Int'l Conf. (eScience 2008). 2008. 640−645. [doi: 10.1109/eScience.2008.167]

[14] Zhang P, Wang GL, Xu XH. A data placement approach for workflow in cloud. Journal of Computer Research and Development, 2013,50(3):636−647 (in Chinese with English abstract).

[15] Zheng P, Cui LZ, Wang HY, Xu M. A data placement strategy for data-intensive applications in cloud. Chinese Journal of Computers, 2010,33(8):1472−1480 (in Chinese with English abstract). [doi: 10.3724/SP.J.1016.2010.01472]

[16] Pandey S, Wu LL, Guru SM, Buyya R. A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments. In: Proc. of the 24th IEEE Int'l Conf. on Advanced Information Networking and Applications (AINA). 2010. 400−407. [doi: 10.1109/AINA.2010.31]

[17] Özsu MT, Valduriez P. Principles of Distributed Database Systems. 3rd ed., Berlin: Springer Science & Business Media, 2011. 31−41. [doi: 10.1007/978-1-4419-8834-8]

[18] Deng KF, Ren KJ, Song JQ, Yuan D, Xiang Y, Chen JJ. A clustering based coscheduling strategy for efficient scientific workflow execution in cloud computing. Concurrency and Computation-Practice & Experience, 2013,25(18):2523−2539. [doi: 10.1002/cpe.3084]

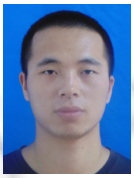
[19] McCormick WT, White T. Problem decomposition and data reorganization by a clustering technique. Operations Research, 1972, 20(5):993−1009. [doi: 10.1287/opre.20.5.993]

附中文参考文献:

[14] 张鹏,王桂玲,徐学辉.云计算环境下适于工作流的数据布局方法.计算机研究与发展,2013,50(3):636−647.

[15] 郑湃,崔立真,王海洋,徐猛.云计算环境下面向数据密集型应用的数据布局策略与方法.计算机学报,2010,33(8):1472−1480. [doi: 10.3724/SP.J.1016.2010.01472]

李学俊(1976－),男,安徽金寨人,博士,副教授,CCF 会员,主要研究领域为云计算,智能软件.

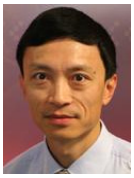


程慧敏(1990－),男,硕士生,主要研究领域为工作流系统.

吴洋(1991－),男,硕士生,主要研究领域为云计算.

朱二周(1981－),男, 博士,讲师,主要研究领域为虚拟化,云计算.

刘晓(1982－),男,博士,副教授,CCF 会员,主要研究领域为软件工程,云计算.



杨耘(1963－),男,博士,教授,博士生导师,主要研究领域为软件技术,云计算.