软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2017,28(3):544−562 [doi: 10.13328/j.cnki.jos.005160]

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

面向多源大数据云端处理的成本最小化方法*

肖文华[1], 包卫东[1], 朱晓敏[1], 邵屹杨[1], 陈 超[1], Jianhong WU[1,2]

[1](国防科学技术大学 信息系统工程重点实验室,湖南 长沙 410073)

[2](Department of Mathematics and Statistics, York University, M3J1P3, Canada)

通讯作者: 包卫东, E-mail: wdbao@nudt.edu.cn

摘 要: 云计算为大数据处理提供了一种强大而高效的解决方案.在此模式下,数据管理者(data manager,简称DM)可以租用多个数据中心实时处理地理分散的数据.然而,由于数据产生的动态性以及资源价格的波动性,将数据迁移至哪些数据中心并提供合适的计算资源来处理它们,成为DM低成本处理多源数据的一大问题.首先,将以上问题转换成联合随机优化问题;然后,利用李雅普诺夫(Lyapunov)优化框架将原问题分解成两个独立的子问题进行求解;最后,基于求解结果设计在线算法.理论分析结果表明:所提算法可不断趋近线下最优解,并能够保证数据处理时延.在WorldCup98和Youtube数据集上的实验验证了理论分析结果的正确性以及该方法的优越性.

关键词: 大数据处理;多数据中心;数据管理;数据迁移;资源供给

中图法分类号: TP311

中文引用格式: 肖文华,包卫东,朱晓敏,邵屹杨,陈超,Wu JH.面向多源大数据云端处理的成本最小化方法.软件学报,2017,28(3):544−562. http://www.jos.org.cn/1000-9825/5160.htm

英文引用格式: Xiao WH, Bao WD, Zhu XM, Shao YY, Chen C, Wu JH. Cost minimization method for multi-source big data processing in clouds. Ruan Jian Xue Bao/Journal of Software, 2017,28(3):544−562 (in Chinese). http://www.jos.org.cn/1000-9825/5160.htm

Cost Minimization Method for Multi-Source Big Data Processing in Clouds

XIAO Wen-Hua[1], BAO Wei-Dong[1], ZHU Xiao-Min[1], SHAO Yi-Yang[1], CHEN Chao[1], Jianhong Wu[1,2]

[1](Science and Technology on Information System Engineering Laboratory, National University of Defense Technology, Changsha 410073, China)

[2](Department of Mathematics and Statistics, York University, M3J1P3, Canada)

Abstract: Cloud computing has shown to provide a cost-effective and powerful platform for big data processing. Under this paradigm, data manager (DM) usually rents geographically distributed datacenters to process their geographically dispersed data set, concerning its convenience and economy. Usually, the data sets are dynamically generated and the resource pricing varies over time, which make it a critical issue of cost effectiveness to move the data from different geographic locations to different datacenters while providing suitable computation resources for processing. In this paper, a pertinent joint stochastic optimization problem is firstly formulated, and then the problem is decoupled into two independent subproblems with efficient solutions via Lyapunov framework. Next, an online algorithm based on the solutions is developed. Theoretical analysis show that the proposed online algorithm can produce a solution which is

* 基金项目: 国家自然科学基金(61572511); 海外及港澳学者合作研究基金(11428101); 国防科学技术大学科研计划(ZK16-03-57, ZK16-03-09); 西南电子电信技术研究室公开课题(2015014)

Foundation item: National Natural Science Foundation of China (61572511); The Overseas, Hong Kong & Macau Scholars Collaborated Research Fund of China (11428101); Scientific Research Project of National University of Defense Technology (ZK 16-03-57, ZK16-03-09); The Southwest Electron&Telecom Technology Insitute Open Project (2015014)

收稿时间: 2016-07-26; 修改时间: 2016-09-14; 采用时间: 2016-11-01; jos在线出版时间: 2016-11-29

CNKI网络优先出版: 2016-11-29 13:35:02, http://www.cnki.net/kcms/detail/11.2560.TP.20161129.1335.007.html

arbitrarily close to the offline optimal solution while minimizing the data processing delays. Experiments on WorldCup98 and Youtube dataset validate the proposed algorithms and demonstrate the superiority of the new approach.

Key words: big data processing; multi datacenters; data management; data moving; resource provisioning

当前,人类社会数据产生模式呈现出高速、大容量的特性.例如,Youtube 每天产生近 40 亿条的视频查看记录以及近 43 万小时的新视频[1],天文望远镜项目 SKA(square kilometer array)每秒产生近 40GB 的数据量[2].随着社会各行各业数据量的快速增长,其蕴含的巨大潜在价值可以通过大数据挖掘和处理提取利用.数据分析已逐步在各行业发挥着越来越重要的作用,比如金融分析、社交网站、天文望远镜服务等.就社交网站来讲,其可以通过分析网站历史记录(如点击记录、活动记录等)揭示用户使用模式以及潜在的关系,以检测社会热点事件或为其市场决策服务.对于如此大规模数据,将其长时间存储成本巨大,因此,进行高效的、近实时的数据处理必不可少.然而,由于数据呈现出地理分散性、容量巨大性以及结构复杂性等特点,利用普通机器对其进行近实时处理和分析往往不可行.

云计算按照即付即用的模式运行,使得用户能够根据自身所需动态调整租用的资源,并且具有高性能以及高容错特性,为大数据处理提供了一种高效而经济的解决方案.在云计算模式下,如何对数据与云资源进行有效管理,为数据管理者降低数据处理成本至关重要.其中,最为重要的问题是:

(1) 如何动态地将不同位置的实时产生的大规模数据分配至地理分布的数据中心?

(2) 需要在这些数据中心中提供多少计算资源以保证服务质量,同时又最小化运行费用?

由于数据产生的动态性、多源性以及资源价格的动态性,使得上述问题变得极具挑战.

当前,对大数据的研究主要集中在不同类型数据的高速并行处理(如针对批量数据处理的 MapReduce[3]框架、针对交互式数据的 Spark[4]系统、针对流式数据处理的 Dremel[5]系统以及针对图数据的 Pregel[6]系统)、大数据分析应用(如个性化推荐[7]、软件分类[8]、基因选择[9])以及大数据处理基础技术[10−14]等方面,但将大规模数据传输到云端并对其数据与资源进行管理的研究很少.目前,为了解决数据迁移问题,常常采用一些简单低效的方法.例如,将数据拷贝至大容量的硬盘中再进行物理运输,甚至直接将整台机器搬运到数据中心等[15,16].这些方法不仅会产生不可容忍的数据处理延迟,而且在运输过程中硬盘会毁坏,具有极大的安全隐患.也有实际项目实现了在数据中心之间根据需要自动复制和传送数据[17,18],但主要聚焦数据的业务需求,未考虑数据处理所需要的资源.

为此,本文对多源大数据云端处理的数据和资源管理问题进行研究,以优化大数据云端处理的成本,提高服务质量.基于此,首先将大数据云端处理的数据迁移和资源供给问题转化为联合随机优化问题;然后,应用李雅普诺夫(Lyapunov)优化技术对模型进行求解并设计相应的在线决策算法.该算法不需要预测系统的未来状态,仅仅基于系统的当前状态做出决定.本文的主要贡献如下:

(1) 提出了一种跨数据中心联合优化数据迁移以及资源供给统一模型,考虑了多数据源数据产生的动态性以及云端不同虚拟机类型及其价格的动态性;

(2) 通过利用李雅普诺夫优化技术解决联合随机优化问题.基于所推导的解析解设计了相应的高效在线决策算法,该算法能够同时做出数据迁移以及资源供给决策并能分布式实现;

(3) 通过理论推导和大量实验对算法性能进行了分析和评估.理论结果表明:该算法可通过调整参数不断趋近于理论最优解,并且能在预设的延迟内完成数据处理任务.实验结果验证了理论分析结果的正确性,并显示了算法的优越性.

本文第 1 节介绍相关研究工作.第 2 节对问题进行描述和形式化建模.第 3 节采用李雅普诺夫优化理论设计在线求解算法.第 4 节对所提算法的性能进行理论分析.第 5 节通过大量实验验证算法的有效性和优越性.最后一节对全文进行总结.

1 相关工作

目前,利用多数据中心进行数据处理和存储日趋流行.就实际系统来讲,当前已有多个商业公司如Facebook,Google,HP,Cisco 开始管理多个地理分布的数据中心,以支持其数据处理业务.为了使 Hadoop 支持地理分布的数据存储,Facebook 开发了一项 Prism 项目[17],其在 Hadoop 集群中增加一层抽象逻辑,聚焦于容错与负载均衡.Google 以一种分布式方式部署了其数据库系统 Spanner[18],使得数据能在多数据中心之间自动迁移.HP[19]与 Cisco[20]也在管理其地理分布的数据中心上付诸实践,主要通过优化数据中心之间的数据链路层实现.然而,当前方法受限于其传输依赖,具有复杂性高而弹性缺失的缺点.

就具体技术来讲,已经有学者研究过将大规模数据集迁移至云中.Cho 等人研究了多数据源向单点数据汇传输的问题,采用网络与物理运输结合的方式,当网络带宽不足时,可以将部分数据通过物理运输方式转移.文献[2]研究了延迟截止期内如何寻找最佳传输策略以最小化传输花费的问题,作者将问题建模成网络流图模型,然后基于整数规划算法求解.文献[21]研究的问题背景与文献[22]相同,但将问题目标转换为在花费预算一定的情况下,如何寻找最优决策以最小化延迟.然而,这些研究将物理传输作为数据转移的一种重要方式,无法避免可能带来的数据损坏和安全性问题.另外,所研究的问题属于静态问题(各数据源数据量固定、网络环境稳定以及物理传输链路固定),不同于本文研究的数据以及资源的动态性.文献[23]研究了如何将不同位置动态产生的数据传输至云中的问题,考虑迁移代价、计算以及存储费用,并以最小化花费为目标提出了在线算法.针对可延迟的数据处理,文献[24]在考虑百分比流量收费模型(percentile charge model)的基础上又研究了在保证数据处理截止期的前提下,如何最小化费用的问题.这些工作的共同点在于主要关注数据的迁移,且假设处理中心的资源无限且稳定.不同于这些工作,本文认为数据和数据中心资源都处于动态变化状态,且数据中心资源有限,因此在研究数据迁移的同时还考虑资源动态供给,并且利用优化方法设计在线算法,在保证延迟的前提下,又能最小化费用.

基于资源虚拟化技术,许多学者对云中资源动态供给问题进行了研究.为降低能耗,Liu 等人提出了支持虚拟机迁移和虚拟机放置优化的调度策略,且能够实现系统更高的自治程度[25].Petrucci 等人考虑了开启和关闭服务器的费用消耗,利用虚拟化技术进行资源整合,并基于此提出了动态调度算法以优化集群能耗[26].为应对突发闪聚请求,文献[27]提出了多媒体云中自适应请求分配和服务能力缩放机制.然而,这些研究往往需要确定的机制来预测未来的工作负载.与此不同,我们采用了李雅普诺夫优化框架,所设计的在线算法不依赖于任何未来数据处理的负载信息.

李雅普诺夫优化技术是近年发展起来的新兴优化技术,主要思想是将长时间序列内的复杂优化问题转化为单时间序列内的漂移-惩罚最小化问题以获得次优解,其独特的优势就是不需要关于未来的任何信息.文献[28]首次提出应用李雅普诺夫优化技术解决网络的稳定性问题,然后被引入到云计算中以解决系统长时间运行下的优化问题(如负载均衡问题[29,30]、效能权衡问题[31,32]、成本优化问题[33]、节能问题[29,30,34]、定价问题[35]),从而达到降低运营成本、节省能耗、提高服务质量的目的.Urgaonkar 等人首先将此技术应用于云计算中,并利用此方法以解决虚拟化数据中心的请求准许和动态资源分配问题[29].Yao 等人利用此技术优化分布式数据中心的负载分配以及计算资源的供给问题,以减小数据中心的用电消耗[34],而且将传统李雅普诺夫模型由单时间尺度拓展至双时间尺度.Wu 等人将此技术应用于云视频流服务中的请求分配与计算资源缩放问题,在降低运营成本的同时,保证用户请求的低延迟响应[31].Zhao 等人考虑到云中多数据中心常常需要根据用户需求和资源状态提供动态定价的机制以最大化其利润,利用此技术解决云资源的动态定价问题[35].Zhu 等人通过建立效用与能耗模型,以权衡系统效用与能耗为目标,利用此方法动态决定各数据中心的虚拟机开启量以及请求准入量以解决效用与能耗的权衡问题[32].Niu 等人针对大型网站存在的请求闪聚问题提出利用私有云和公有云的混合策略,在负载大时,通过租借公有云的资源满足用户需求,并通过李雅普诺夫技术解决如何根据负载变化动态地向公有云租借合适的资源以应对请求闪聚问题[33].考虑到不同区域碳排放标准以及速率的不同,文献[30]通过此优化方法对负载分配、数据中心容量缩放以及 CPU 速率调整问题同时进行决策,在降低能耗的同时,保证碳排放量满足当地要求.这些工作之所以取得很好的效果,主要在于李雅普诺夫优化框架具有对未知状态不敏

感的特性.然而,这些工作采用传统的李雅普诺夫框架,主要在目标函数和惩罚函数之间进行权衡,在任务处理时效性上表现出一种尽力而为的特性,因而算法不能保证特定的处理延时.不同于以上研究,本文聚焦的是多源大数据云端实时处理的数据迁移和资源管理优化问题.据我们所知,还尚未有利用李雅普诺夫解决大数据云中处理相关问题的文献.而且,通过引入延迟容忍队列设计在线算法,使得数据能在预设的延迟内完成,从而保证数据处理的实效性.

总之,与已有研究相比,本文创新性表现在以下几个方面.

1) 面向动态多源大数据云端近实时、高效、低成本处理,在考虑数据动态、资源价格动态以及延迟保证的情况下,以最小化带宽费用、存储费用、计算费用以及延迟费用为目标,对其中的动态数据迁移和资源供给的问题,建模成长时间运行的联合随机整数优化问题,设计了可分布式执行的在线算法;

2) 采用李雅普诺夫框架解决长时间内数据迁移和资源供给联合优化问题,其不需要对未来数据量进行预测,这与需要特定预测机制的传统方法(如文献[22–24])有显著不同.利用此方法,虽然数学形式复杂,但最终决策时能够获得解析解,保证系统具有实时性.理论分析显示:该算法在提高求解效率的同时,可无限趋近于线下最优解;

3) 本文通过引入延迟容忍队列,保证了数据在特定的延迟内完成处理,克服了大多数基于李雅普诺夫的工作对延迟仅仅尽力而为的缺点.

2 问题建模

本节首先对问题进行了描述,然后将问题形式化为数学优化模型.

2.1 问题描述

本文所考虑的系统架构如图 1 所示:数据管理者(例如社交网站 Facebook)管理着多个分布在各地的不断产生海量数据的数据节点,为充分挖掘数据的价值,管理者需将这些地理分散的数据进行实时处理.为避免大量建设基础设施所需的成本,数据管理者通过租用公有云资源(如 Amazon EC2)处理数据,然后将数据处理结果形成各种应用向用户提供服务(如热点事件监测、用户行为分析、广告投放等).

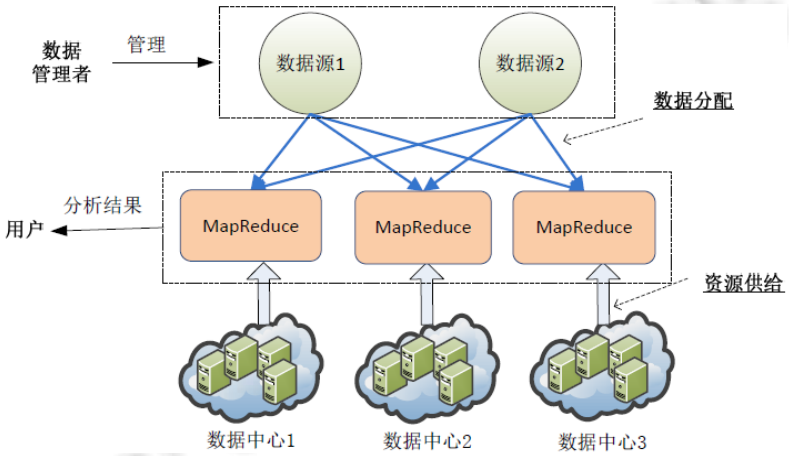

Fig.1 Example of system architecture

图 1 系统架构示意

具体来讲,数据管理者将数据源与数据中心相连并将数据分析应用部署在云端.各数据源一旦产生数据,就被迁移到相应的数据中心并进行处理.各数据中心采用分布式的处理模型(如 MapReduce)进行数据处理,因而所租用的每个虚拟机就可当做 MapReduce 中的一个节点.在此背景下,数据管理者面临的问题是:如何决策各数据源到各数据中心的大规模数据迁移以及在各数据中心租用多少量的虚拟机资源,从而在保证数据高效处理

的同时最小化系统总成本.显然,由于数据的动态性、资源价格在不同数据中心的异构性以及各链路带宽资源以及各数据中心负载的波动性,采用简单的规则(如将数据仅迁移至最近的数据中心或迁移至价格最小的数据中心)往往不是最优的决策(对比实验部分会验证).针对此问题,本文致力于在优化系统长时间运行下,数据处理和资源供给在各数据中心的安排.在系统运行过程中,数据管理者监视着数据中心的状态(比如虚拟机的价格、数据中心的负载状态、网络状态),并以此为知识,决定转移到每个数据中心的数据量以及从每个数据中心租用的资源量,从而达到运行成本最小化的目的.最后,数据中心在数据运行和分析后,将分析结果形成大数据应用向用户提供服务.

形式化地,本问题考虑地理分布的数据中心集合$\mathcal{D}$,其总数为 $D=|\mathcal{D}|$,取值为 $d(1\leqslant d\leqslant D)$.各数据中心配置有不同类型的虚拟机$\mathcal{K}$(具有大小 $K=|\mathcal{K}|$),每类虚拟机有不同的 CPU 和内存配置,并设 k 类虚拟机的能力为 v_k,表示该类虚拟机处理数据的速率.数据处理速度与 MapReduce 具体应用有关,不同的数据处理具有不同的速率.数据管理者管理着 $R=|\mathcal{R}|$个数据源(表示为集合$\mathcal{R}$),且各数据源(取值为 $r,1\leqslant r\leqslant R$)动态产生需要处理的数据.为此,任何数据源的数据可通过虚拟专用网(virtual private network,简称 VPN)移动到其所租用的数据中心来进行分析.为模拟真实场景,我们假设从数据源 r 到数据中心 d 上的 VPN 连接(r,d)的带宽 B_r^d 是有限的,并且是系统瓶颈之一.此外,每个地理位置生成的数据量是独立的,每个数据中心的资源价格(例如虚拟机、存储)是不同的,并且随时间变化.

该系统依照时间序列运行,划分为 $t=0,1,\ldots,T$.在每个时间序列中,数据管理者需要决定从数据源 r 移动多少数据到数据中心 d 以及每个数据中心租用多少资源来支持数据处理.所优化的目标为最小化云端对大数据分析的总成本,并且能够保证在长时间运行中数据处理的延迟.为了便于参考,一些重要的符号在表 1 中列出.

Table 1 Important notations

表 1 重要符号

$\mathcal{D}$	数据中心的集合
$\mathcal{R}$	数据源集合
$\mathcal{K}$	虚拟机类型的集合
$a_r(t)$	在 t 时刻从数据源 r 产生的数据量
$A_{\max}^r$	数据源 r 产生的最大数据量
$\lambda_r^d(t)$	t 时刻从数据源 r 移动到数据中心 d 的数据量
$N_d^{k,\max}$	数据中心 d 中可租用的 k 类型虚拟机数量
$n_d^k(t)$	t 时刻从数据中心 d 中租用的 k 类型虚拟机数量
$p_d^k(t)$	t 时刻数据中心 d 中 k 类型虚拟机的价格
s_d	数据中心 d 中存储的价格
b_r^d	从数据源 r 到数据中心 d 的带宽价格
v_k	k 类型虚拟机的数据处理速率
ε_d	$H_d(t)$中控制队列延迟的预设常数
l	数据处理的最大延迟
$H_d(t)$	t 时刻数据中心 d 上未处理的数据
$Z_d(t)$	与 $H_d(t)$关联的来保证延迟的虚拟队列

2.2 问题形式化

本节首先公式化系统所需的费用,然后对所优化的问题进行数据建模.

如前所述,系统以时隙方式运行,不同数据源、不同时隙的数据动态地生成.设 $a_r(t)$为 t 时刻数据源 r 生成的数据量.由于从任意数据源生成的数据可移动到任意数据中心进行处理,我们设 $\lambda_r^d(t)$ 为在 t 时刻从数据源 r 移动到数据中心 d 的数据量, $A_{\max}^r$ 为数据源 r 产生的最大数据量.则有:

$$a_r(t)\leqslant A_r^{\max},\forall r,t\in[1,T] \tag{1}$$

$$a_r(t)=\sum_{d\in\mathcal{D}}\lambda_r^d(t),\forall r,t\in[1,T] \tag{2}$$

数据管理者的目标是:通过优化分配给不同数据中心的数据量和数据中心所需的资源,以使系统中产生的总成本最小化.基于此,本文主要考虑以下成本要素:带宽成本、延迟成本、储存成本和计算成本,各成本详细定义如下.

(1) 一般情况下,由于不同 VPN 属于不同的互联网服务提供商,其带宽价格各不相同.令 b_r^d 是从数据源 $r\in\mathcal{R}$传输 1GB 数据到数据中心 $d\in\mathcal{D}$的价格,则 t 时刻带宽的总费用可定义为

$$C_b(t)\triangleq\sum_{d\in\mathcal{D}}\sum_{r\in\mathcal{R}}\lambda_r^d(t)\cdot b_r^d \tag{3}$$

(2) 由于数据规模的庞大,数据的存储成本也是影响数据中心选择的重要因素之一.令 s_d 为单时隙内数据中心 $d\in\mathcal{D}$上储存 1GB 数据所需要的成本,则 t 时刻系统产生的储存总成本为

$$C_s(t)\triangleq\sum_{d\in\mathcal{D}}\sum_{r\in\mathcal{R}}\lambda_r^d(t)\cdot s_d \tag{4}$$

(3) 由于各云服务提供商通常采用动态定价机制,虚拟机价格往往随着时间不断变化(如 Amazon EC2 虚拟机实例),因而,从数据中心租用的虚拟机数量对系统的总成本和服务质量有重要影响.令 $n_d^k(t)$ 为 t 时刻从数据中心 d 中租用的 k 类型虚拟机数量,令 $p_d^k(t)$ 为 t 时刻数据中心 d 中 k 类型的虚拟机价格,则数据处理所需要的计算成本为

$$C_p(t)\triangleq\sum_{d\in\mathcal{D}}\sum_{k\in\mathcal{K}}n_d^k(t)\cdot p_d^k(t) \tag{5}$$

(4) 考虑到数据源与数据中心分布在不同地理位置,本文将延迟作为数据处理需要考虑的重要性能指标,数据迁移时要尽可能减小延迟造成的影响.由于数据传输的距离较长且传输量很大,本文主要考虑传输数据到数据中心所造成的延迟.令 L_r^d 为数据源 $r\in\mathcal{R}$传输数据到数据中心 $d\in\mathcal{D}$的延迟.为简单起见,本文认为,延迟主要由地理距离等因素决定,实际系统中,其可以通过简单的命令如 Ping 来获得.如文献[15]所建议,本文将延迟转换为经济成本.因此,可以定义延迟为

$$C_l(t)\triangleq\sum_{d\in\mathcal{D}}\sum_{r\in\mathcal{R}}\alpha\cdot\lambda_r^d(t)\cdot L_r^d \tag{6}$$

其中,α是将延迟转换为经济成本的权重系数.基于以上的成本公式,可以导出系统中产生的总成本为

$$C(t)=C_p(t)+C_s(t)+C_b(t)+C_l(t) \tag{7}$$

基于以上各费用成本的定义,根据本文所研究的目标,最小化时间段$[0,T]$内数据迁移和处理的时间平均成本可以形式化为

$$\mathbf{P1}.\min:\lim_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T-1}\mathbb{E}\{C(t)\} \tag{8}$$

$$\text{s.t. } a_r(t)\leqslant A_r^{\max},\forall r,t\in[1,T] \tag{9}$$

$$a_r(t)=\sum_{d\in\mathcal{D}}\lambda_r^d(t),\forall r,t\in[1,T] \tag{10}$$

$$0\leqslant n_d^k(t)\leqslant N_d^{k,\max},\forall d,\forall k,t\in[1,T] \tag{11}$$

$$n_d^k(t)\in Z^+\cup 0,\forall d,\forall k,t\in[1,T] \tag{12}$$

其中,约束(10)是为了确保在单时隙内分配给各数据中心数据的总和等于在该时刻产生的总数据量,约束(11)确保了所需的虚拟机数量不超过数据中心可以提供的范围.从问题 **P1** 表达来看,由于数据生成是未知且动态的,资源变量 $n_d^k(t)$ 是整数型,因此,以上问题是一个约束随机整数优化问题.本文的目标是:在长期运行状态下,通过优化分配给每个数据中心的数据以及数据中心租用的虚拟机的数量,以使长期数据处理平均成本最小化.为了处理此问题,本文采用了优化技术——李雅普诺夫优化框架来对问题进行求解.

3 在线算法设计

本节利用李雅普诺夫优化理论来设计在线控制算法.该方法的突出特点是不需要有关未来负载的任何信息,通过贪婪地最小化在每个时间序列中的漂移惩罚,理论上可得到一个任意接近线下最优解的次优方案.根据标准的李雅普诺夫优化框架理论[36],我们首先将问题 **P1** 变换为最小化李雅普诺夫漂移惩罚项的优化问题,然后设计相应的在线算法.

1) 问题转换.

令 $H_d(t)$为 t 时间序列上数据中心 d 中未处理的数据量.首先,我们定义 $H_d(t)$=0,则队列 $H_d(t)$的演化可以描述如下:

$$H_d(t+1)=\max\left[H_d(t)-\sum_{k\in\mathcal{K}}n_d^k(t)\cdot v_k,0\right]+\sum_{r\in\mathcal{R}}\lambda_r^d(t) \tag{13}$$

上述队列的更新规则意味着所处理的数据量为 $\sum_{k\in\mathcal{K}}n_d^k(t)\cdot v_k$,新到达的数据量为 $\sum_{r\in\mathcal{R}}\lambda_r^d(t)$.为保证队列 $H_d(t)$, $\forall d\in\mathcal{D}$在最坏情况下的延迟处于最大工作负载延迟 l 内,我们根据文献[37]中的ε-持久服务策略设计了相关虚拟队列 $Z_d(t)$(可视为延迟容忍队列).其中,虚拟队列 $Z_d(t)$的负载初始化为 $Z_d(t)$=0,且更新规则如下:

$$Z_d(t+1)=\max\left[Z_d(t)+1_{H_d(t)>0}\left(\varepsilon_d-\sum_{k\in\mathcal{K}}n_d^k(t)\cdot v_k\right)-1_{H_d(t)=0}\sum_{k\in\mathcal{K}}N_d^{k,\max}\cdot v_k,0\right] \tag{14}$$

其中,指示函数 $1_{H_d(t)>0}$ 表示当 $H_d(t)>0$ 时等于 1,否则等于 0;同样地,$1_{H_d(t)=0}$ 表示当 $H_d(t)=0$ 时为 1,否则为 0;ε_d 是预设常数,用来控制队列延迟的范围.由此可以证明:若所提算法能够保证队列 $H_d(t)$和 $Z_d(t)$长时间的稳定,则所有数据都可以在至多 l 个时隙延迟内得到处理.并且 l 可设置为 $l=[H_d^{\max}+Z_d^{\max}/\varepsilon_d]$,其中, $H_d^{\max}$ 和 $Z_d^{\max}$ 分别是队列 $H_d(t)$和 $Z_d(t)$的上限,详情请参阅定理 2.

令 $\boldsymbol{Z}(t)=(Z_d(t)),\boldsymbol{H}(t)=(H_d(t)),\forall d\in\mathcal{D}$分别表示虚拟队列和实际队列的矩阵,可以用 $\Theta(t)=[\boldsymbol{H}(t),\boldsymbol{Z}(t)]$来表示实际队列和虚拟队列的联合矩阵.据李雅普诺夫框架[36],我们定义李雅普诺夫函数如下:

$$L(\Theta(t))=\frac{1}{2}\sum_{d\in\mathcal{D}}\{Z_d(t)^2+H_d(t)^2\} \tag{15}$$

其中,$L(\Theta(t))$为系统中负载积压的度量,则单时隙的李雅普诺夫漂移函数则可定义为

$$\Delta(\Theta(t))=\mathbb{E}\{L(\Theta(t+1))-L(\Theta(t))|\Theta(t)\} \tag{16}$$

为在保证系统队列稳定的同时还最小化系统所产生的花费,则李雅普诺夫漂移-惩罚项可以在公式(16)漂移函数中增加系统总成本函数获得,即:

$$\Delta(\Theta(t))+V\cdot\mathbb{E}\{C(t)|\Theta(t)\} \tag{17}$$

其中,V 为非负参数,它可以在系统稳定性和成本之间进行折衷.V 越大,系统产生的成本就越小;反之,成本就越大.因此,原来的问题 **P1** 就变成了下面的问题 **P2**:

$$\textbf{P2}.\min\ (17) \tag{18}$$

$$\text{s.t.}\quad (9)(10)(11)(12) \tag{19}$$

为了解决 **P2**,本文不直接最小化漂移-惩罚函数,而是致力于最小化它的上界.然而,理论证明,此方式并不破坏算法的最优性和性能[36].因此,求解 **P2** 的关键是找到其上界.通过理论推导可证明,公式(17)的界为

$$\begin{aligned}\Delta(\Theta(t))+V\cdot\mathbb{E}\left\{\lim_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T-1}C(t)\middle|\Theta(t)\right\}\leqslant B+\mathbb{E}\left\{\sum_{d\in\mathcal{D}}\sum_{k\in\mathcal{K}}n_d^k(t)\cdot(Vp_d^k(t)-H_d^k(t)v_k-Z_d^k(t)v_k)\middle|\Theta(t)\right\}+\\ \mathbb{E}\left\{\sum_{d\in\mathcal{D}}\sum_{r\in\mathcal{R}}\lambda_d^r(t)\cdot(Vs_d+Vb_r^d+VL_r^d+H_d(t))\middle|\Theta(t)\right\}\end{aligned} \tag{20}$$

其中, $B=\frac{1}{2}\sum_{d\in\mathcal{D}}\left\{2\left(\sum_{k\in\mathcal{K}}N_d^{k,\max}v_k\right)^2+(\varepsilon_d^k)^2+\left(\sum_{r\in\mathcal{R}}A_{\max}^r\right)\right\}$.详细的证明请见附录 A.

2) 在线算法的设计.

通过仔细研究不等式(20)的右边,发现该优化问题可以等价地分解成两个子问题,即数据分配问题和资源供应问题.求解以上两个子问题的细节如下所述.

(1) 数据迁移

为最小化公式(20)的右边,通过观察变量之间的关系,其中与数据分配相关的部分可被提取为

$$\min\ \mathbb{E}\left\{\sum_{d\in\mathcal{D}}\sum_{r\in\mathcal{R}}\lambda_d^r(t)\cdot(Vs_d+Vb_r^d+V\alpha L_r^d+H_d(t))\middle|\Theta(t)\right\} \tag{21}$$

此外,由于各数据源的数据是独立生成的,公式(21)所述的多数据源整体优化方式可以分别在各数据源独立执行.考虑 t 时刻数据源 r 上数据分配,则问题转化为解决如下问题:

$$\begin{aligned}&\min\ \sum_{d\in\mathcal{D}}\lambda_d^r(t)[Vs_d+Vb_r^d+V\alpha L_r^d+H_d(t)]\\&\text{s.t.}\ \ (9)(10)\end{aligned} \tag{22}$$

事实上,上述问题是一个广义的最小权重问题,从数据源 r 迁移到数据中心 d 的权重为 $\lambda_d^r(t)$,它与数据积压 $H_d(t)$、带宽成本 b_r^d 、储存成本 s_d、延迟成本 L_r^d 有关.通过使用线性规划理论,我们可以求得以下解决方案:

$$\lambda_r^d(t)=\begin{cases}a_r(t), & d=d^*\\0, & \text{else}\end{cases} \tag{23}$$

其中, $d^*=\min_d[Vs_d+Vb_d+V\alpha L_r^d+H_d(t)]$.显然,t 时刻算法倾向于将数据源 r 产生的数据迁移至该时刻具有最短任务队列和最小运行成本的数据中心进行处理.

(2) 资源配置

若去掉公式(20)右边的常数项 B,则与变量 $n_d^k(t)$ 相关的部分可以被认为是资源供应问题.因此,我们可以通过解决如下问题得到虚拟机最优供应策略:

$$\begin{aligned}&\min\ \mathbb{E}\left\{\sum_{d\in\mathcal{D}}\sum_{k\in\mathcal{K}}n_d^k(t)\cdot(Vp_d^k(t)-H_d(t)v_k-Z_d(t)v_k)\middle|\Theta(t)\right\}\\&\text{s.t.}\ \ (11)(12)\end{aligned} \tag{24}$$

同理,由于各数据中心中的资源供给是独立的,与数据分配问题相似,公式(23)可以在每个数据中心间分布地求解.因而对于单个数据中心 d,资源供应问题可以进一步改写为

$$\begin{aligned}&\min\ \mathbb{E}\left\{\sum_{k\in\mathcal{K}}n_d^k(t)\cdot(Vp_d^k(t)-H_d(t)v_k-Z_d(t)v_k)\middle|\Theta(t)\right\}\\&\text{s.t.}\ \ (11)(12)\end{aligned} \tag{25}$$

易得上述线性问题的解为

$$n_d^k(t)=\begin{cases}N_d^{k,\max}, & \text{if } H_d(t)+Z_d(t)>\dfrac{Vp_d^k(t)}{v_k}\\0, & \text{if } H_d(t)+Z_d(t)\leqslant\dfrac{Vp_d^k(t)}{v_k}\end{cases} \tag{26}$$

上述解决方案表明:当 t 时刻 k 类虚拟机的价格 $p_d^k(t)$ 越小,而其虚拟机容量 v_k 越大时,k 类型的虚拟机有更大可能将被租用.

至此,通过利用李雅普诺夫框架对原问题进行转化,长时间内数据迁移和资源供给的成本最小化问题得到有效求解.以上简单解决方案有助于在真实系统中在线部署所提算法,其在线算法的细节见算法 1.

算法 1. 在线算法程序.

1 Input:

2 $H_d(t),Z_d(t),a_r(t),v_k,b_r^d,N_d^{k,\max},A_{\max}^r,p_d^k(t),V,\alpha(\forall d\in\mathcal{D},r\in\mathcal{R},k\in\mathcal{K})$;

3 Output:

4 $n_d^k(t),\lambda_d^r(t)(\forall d\in\mathcal{D},r\in\mathcal{R},k\in\mathcal{K})$.

5 资源供给:

6 **for each** 数据中心 $d\in\mathcal{D}$,虚拟机类型 $k\in\mathcal{K}$ **do**

7 通过应用公式(26)解决问题(24)得到虚拟机供给策略 ($n_d^k(t)$);

8 数据分配:

9 **for each** 数据源 $r\in\mathcal{R}$,数据中心 $d\in\mathcal{D}$ **do**

10 通过应用公式(23)解决问题(21)得到数据分配策略 ($\lambda_d^r(t)$);

11 根据队列动态等式(13)、等式(14)分别更新队列 $H_d(t),Z_d(t)$.

4 算法性能分析

本节从成本最优性、队列负载上界、数据处理最差延迟及算法复杂度方面对算法 1 进行了理论分析.

定理 1(成本最优). 假设数据生成的速率 $a_r(t),\forall r\in\mathcal{R}$在任意时刻是独立同分布的,对于任意控制参数 $V>0$,所提算法产生的系统费用与最优解产生的费用关系为

$$\limsup_{T\to\infty}\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\{C(t)\}\leqslant C^*+\frac{B}{V} \tag{27}$$

其中,C^*表示平均时间成本的下确界,代表着理论上最佳方案所得费用;常数 B 与公式(20)中的定义相同.证明请见附录 B.

该定理表明,本文算法获得的时间平均成本与线下得到的最优成本的差呈现$\mathcal{O}(1/V)$的关系.特别地,通过控制变量 V,时间平均成本 C 可以任意地接近最优解 C^*.

定理 2(队列负载上界). 假设ε_d满足 $\varepsilon_d<\sum_{k\in K}N_d^{k,\max}v_k$.令 $H_d^{\max}$ 和 $Z_d^{\max}$ 分别为队列 $H_d(t)$和虚拟队列 $Z_d(t)$的上界,则

$$Z_d^{\max}=\frac{Vp_d^{\max}}{v_{\min}}+\varepsilon_d \tag{28}$$

$$H_d^{\max}=\frac{Vp_d^{\max}}{v_{\min}}+\sum_{r\in\mathcal{R}}A_{\max}^r \tag{29}$$

其中, $p_d^{\max}$ 为各虚拟机的最高价格,$v_{\min}$ 是各类虚拟机中的最小容量.证明请见附录 C.

此定理说明,队列的负载大小呈现出$\mathcal{O}(V)$关系.这意味着:为保持队列负载的稳定性,应该选择较小的 V.然而,当减小参数 V,又会导致成本的增大(见公式(26)),因此,系统成本和稳定性具有[$\mathcal{O}(1/V),\mathcal{O}(V)$]的折衷关系.在实际应用中,在给定成本预算情况下,我们可以选择合适的 V 以使系统的稳定性最大化;反之亦然.

定理 3(最差情况延迟). 假设系统以先到先出的机制运行,那么队列 d 中数据运行的最差延迟为

$$l=[H_d^{\max}+Z_d^{\max}/\varepsilon_d] \tag{30}$$

其中,[x]表示在那些大于或等于 x 的数中最小的数. $H_d^{\max}$ 和 $Z_d^{\max}$ 如公式(28)、公式(29)所定义.证明请见附录 D.

此定理说明:无论数据任何时间到达队列 H_d,都可以在 l 个时隙内处理完成.这表明本文所提算法能够保证数据处理的服务质量.此外,在给定系统参数的情况下,由于 $H_d^{\max}$ 和 $Z_d^{\max}$ 固定,我们可以通过选择合适的ε_d 来改变云端数据处理的服务质量.同样地,通过设置不同数据中心的不同参数$\varepsilon_d,d\in\mathcal{D}$,可以为不同的数据中心设置异构的服务质量.

定理 4(算法复杂度). 若假设系统中包含的数据源个数、数据中心个数和虚拟机类型数分别为 R,D,K,则算法 1 的复杂度为$\mathcal{O}(D\times K+D\times R+D)$.

证明:算法 1 主要由 3 部分组成:资源配置问题求解部分、数据迁移求解部分和队列更新部分.据算法 1 易得,资源配置问题部分算法复杂度为$\mathcal{O}(D\times K)$,数据迁移部分算法复杂度为$\mathcal{O}(D\times R)$以及队列更新部分算法复杂

度为$\mathcal{O}(D)$.因此,算法 1 的总复杂度是$\mathcal{O}(D\times K+D\times R+D)$.

5 实验及结果分析

本节利用真实数据集对本文所提算法的有效性进行评估.

5.1 数据集描述

鉴于大规模数据分析对全球性大规模网站(例如 Google,Youtube,Facebook 等)市场决策来说越来越重要,我们采用 1998 年世界杯网站真实数据集 Worldcup98[38]和 Youtube 网站记录数据集[39]来评估本文算法性能.

1998 年世界杯网站在地理分布的 4 个位置部署 30 台服务器(巴黎的 4 台服务器、赫恩登的 10 台服务器、普莱诺的 10 台服务器以及圣克拉拉的 6 台服务器)为全球用户提供服务,Worldcup98 数据集记录了该网站从 4 月 30 日~7 月 26 日的用户访问数据,每次访问的记录都存储在处理请求的服务器上,期间总共产生大约有 10 亿条记录.每条记录包含的详细信息如请求时间、用户 ID、请求内容以及处理请求的服务器 ID.我们提取了 6 月 21~27 日共一周的数据进行实验.为仿真大规模的网站,我们将原始的网站请求量扩大 1 000 倍.每隔 30 分钟对请数进行汇总,并假设每次请求的记录内容为 100KB,则可得如图 2 所示的数据变化图.图中的尖峰部分为晚间比赛阶段;而白天网站访问量则较小,整体呈现出周期变化模式.

Youtube 数据集收集了马萨诸塞大学各分区校园的 Youtube 访问数据,记录内容包含访问时间、用户 IP、访问的 Youtube 服务器、访问内容、内容服务器等信息.本文选取 2008 年 1 月 1 日和 1 月 31 日共两天的数据.数据集中有 3 个 Youtube 服务器 IP,因此,设置数据源数量为 3,其他设置与以上相同.图 3 为其数据变化情况.

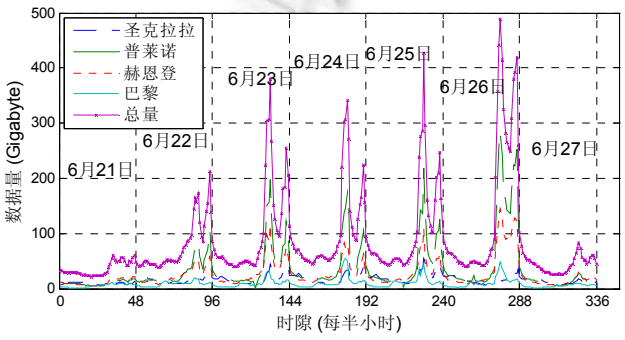

Fig.2 The data volume fluctuation pattern of Worldcup98 dataset

图 2 Worldcup98 数据集数据量随时间变化模式

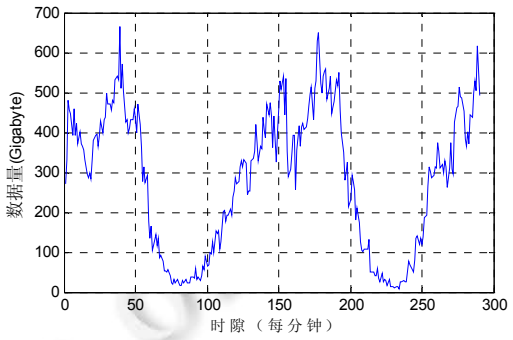

Fig.3 The data volume fluctuation pattern of Youtube dataset

图 3 Youtube 数据集数据量随时间变化模式

5.2 实验设置

在实验中,针对 Worldcup98 数据集,假设模型包含 4 个数据源(与 Worldcup98 数据集中的 4 个数据位置,即位于美国的圣克拉拉、普莱诺、赫恩登以及位于法国的巴黎对应),针对 Youtube 数据集,由于数据集只包含 3 个服务器 IP,假设模型包含 3 个数据源.其他共同设置如下:12 个数据中心(与亚马逊在欧洲和美洲的服务器,即阿什本、达拉斯、洛杉矶、迈阿密、纽瓦克、帕洛阿尔托、西雅图、圣路易斯、阿姆斯特丹、都伯林、法兰克福以及伦敦相对应)[40].虚拟机方面,考虑 Amazon EC2 所提供的 5 种类型的虚拟机实例(即 c3.large,c3.xlarge,c3.2xlarge,c3.4xlarge,c3.8xlarge).数据中心与数据源之间的距离通过在线工具[41]获得,其距离矩阵图详见后文图 6 所示.

为更好发挥模型性能,我们建议模型的一些参数设置如下:与文献[23]相同,采用 RTT(round trip time)测度测量数据源与数据中心的链路延迟,即 RTT(ms)=0.02×distance(km)+5.虚拟机价格与存储价格分别采用亚马逊 Spot instance 价格和 S3 的价格(可以从其网站获取),而通过链路$\langle r,d\rangle$上传数据的单位价格服从[0.1,0.25]美元/GB 的均匀分布.对于某一具体应用来说(如 MapReduce 应用为分年龄段统计每时段用户访问数),由于 Map 与

Reduce 过程相对固定,数据在不同类型的虚拟机中的处理速度也相对稳定,可设其对应的单位核数据处理速率为 100MB/slot(即 c3.large,c3.xlarge,c3.2xlarge,c3.4xlarge,c3.8xlarge 的处理速度分别为 100MB/slot,200MB/slot,400MB/slot,800MB/slot,1600MB/slot).为简单起见,设置数据迁移代价为与数据相关的线性函数.除非特别指出,其他参数默认设置为 V=20,α=0.01,ε_d=1.

基于所提算法及以上参数设置,在 Matlab 中实现并进行仿真实验.以 Worldcup98 和 Youtube 数据集每时刻放大的数据量作为输入,通过所提算法均衡计算成本、存储成本、带宽成本以及延迟成本,做出数据迁移决策以及虚拟机资源的租赁决策,即:根据各个数据源每时刻产生的数据决定每时刻往各个数据中心迁移多少,并提供合适量的不同能力大小的虚拟机数.以此按时隙运行,则产生了一段时间内处理该数据的数据迁移决策和资源供给(购买)决策,从而使得整个时间内的总成本最小.实验均在配置为 Intel i3-3240 CPU,4G RAM 的戴尔 PC 机上进行.为方便读者更深入研究或工程化,本文提供了实验源代码(https://www.researchgate.net/publication/308721003_Cost-aware_Multi-source_Big_Data_Processing_on_Clouds_using_Lyapunov_Technique)作为参考.

5.3 算法的有效性

为验证算法的有效性,本节主要在参数固定情况下,利用 Worldcup98 数据集进行了实验.图 4 显示了系统总花费随时间的变化情况,从图 4 可知,总花费随着数据量的大小而变化(可参考图 2 的数据变化).这说明本文算法能够在没有预测未来负载的情况下自适应动态地调整虚拟机的供给量,以满足不断变化的数据处理需求.图 5 显示了各种虚拟机(即 c3.large,c3.xlarge,c3.2xlarge,c3.4xlarge 以及 c3.8xlarge)费用比随时间变化的对比情况,结果显示:虚拟机性能越强,其花费更高.这是因为我们采取了价格策略为:性能越强的虚拟机单位计算能力价格越低,因此性能越强的虚拟机性价比越高,从而算法会优先选择性能越强的虚拟机(如 c3.8xlarge)进行数据处理.并且在数据处理量比较小的情况下(如时隙 120~140,150~170,200~230 等),性能较大的虚拟机比例会更高.这是因为此时各类型的虚拟机资源充足,性价比高的大容量虚拟机则会优先选择.然而在负载较大情况下,由于资源的缺乏,会租用单位价格高的低性能虚拟机进行补充.

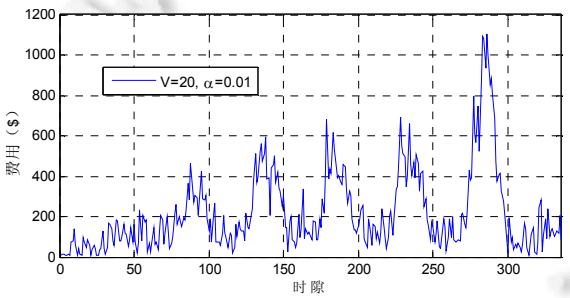

Fig.4 Cost variance along time slots under fixed parameters

图 4 参数固定下总费用随时间的变化

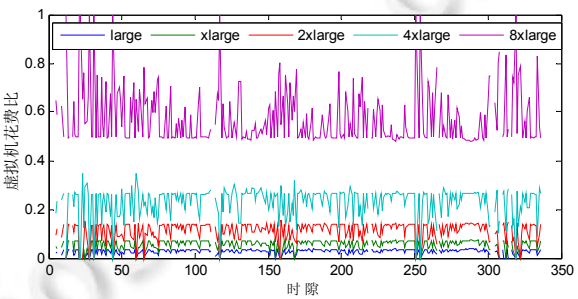

Fig.5 Cost ratio of each type of VM along time slots under fixed parameters

图 5 参数固定下各虚拟机费用随时间变化

更进一步地,为深入剖析算法的特性,对数据分配详细结果进行了展示.结合图 6 和图 7 可知:本文算法结果表现出数据本地化的特性,因为数据倾向于转移至数据源附近的数据中心处理.

特别地,巴黎产生的数据较少转移至北美的数据中心(即阿什本、达拉斯、洛杉矶、迈阿密、纽瓦克、帕洛阿尔托、西雅图、圣路易斯)进行处理,即使北美的价格比欧洲的价格要低.这意味着算法具有避免过大延迟,从而保证数据按时处理的能力.

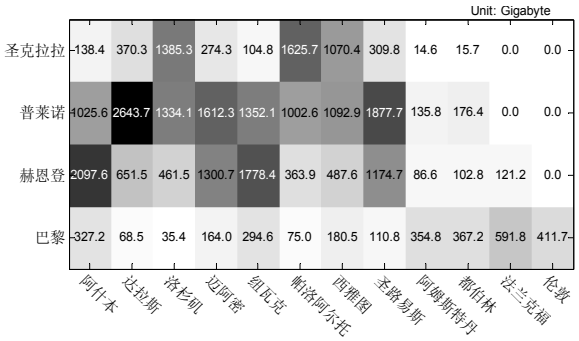

Fig.6 Result of data allocation under fixed parameters

图 6 参数固定下数据分配结果

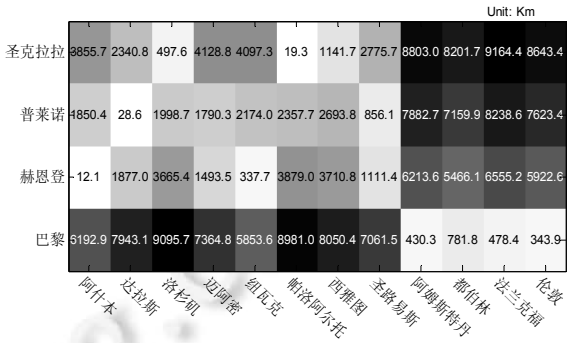

Fig.7 Distance matrix between datasources and datacenters

图 7 各数据源与数据中心的距离矩阵

5.4 参数对性能的影响

本节在 Worldcup98 数据集上通过实验分析了参数 V 对算法性能的影响.图 8 显示了平均花费与队列长度随着参数 V 的变化情况,从图可知:系统产生的时间平均花费随着 V 的递增而降低;并且当 V 足够大时,系统平均费用趋近于稳定最小值.这一结果为我们在部署真实系统时降低费用提供了理论指导.然而随着 V 的增长,负载队列长度也随之增长,队列的增长又会导致数据处理的时延.因此,如何选择合适的 V 以平衡系统总费用以及延迟非常重要.另外,本实验结果还与算法的理论分析结果一致,因而验证了定理 1 推导的正确性.图 9 显示了参数 ε 对平均花费和队列长度的影响,系统产生的费用随着参数 ε 的增大而增大,队列负载呈现出不断递减的趋势.这可能是因为随着 ε 的不断增大,延迟 l 不断减小,因而需要租用更多的资源以更快地进行数据处理,这必然导致系统费用的增加.同理,随着资源的增加,队列负载也会不断减小以降低延迟.

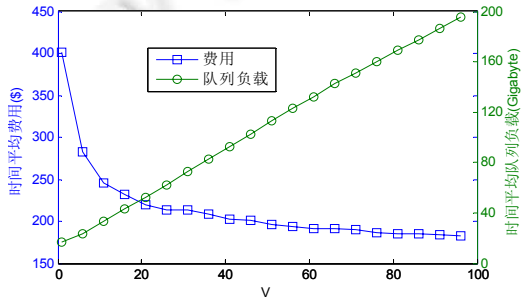

Fig.8 Impact of parameter V

图 8 参数 V 对性能的影响

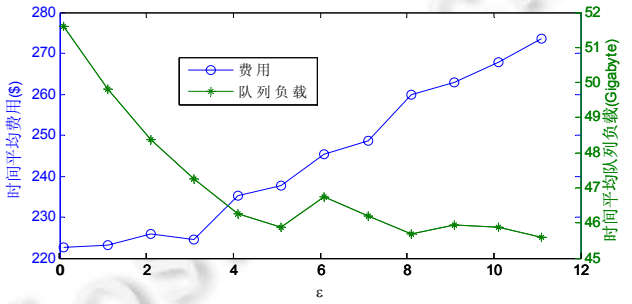

Fig.9 Impact of parameter ε

图 9 参数 ε 对性能的影响

图 10 和图 11 显示了传输延迟代价转换因子 α 对性能的影响,随着 α 的增大,平均花费不断递增,而队列负载不断降低.这是因为 α 增大,意味系统对延迟更敏感,反过来算法能够调整数据迁移以及资源分配使得队列延迟减小.图 11 说明增加 α 基本不影响其他类型花费,进一步说明算法能够优化决策从而最小化各类花费.因而选择这一参数时只要给定了费用的上限,则可以选择合适的参数 α.事实上,由于系统中通常会考虑多目标(比如花费、队列延迟),以上参数仅是权衡各目标的因子,因此在确定某一目标的情况下,则可选择合适的参数使得各目标能够权衡.通常,我们选择曲线交汇点为权衡点,如图 7 中 V=20、图 8 中 ε=4 以及图 9 中 α=0.02.

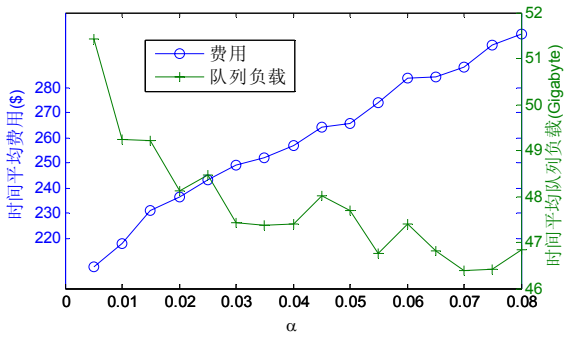

Fig.10 Impact of parameter

图 10 参数α对性能的影响

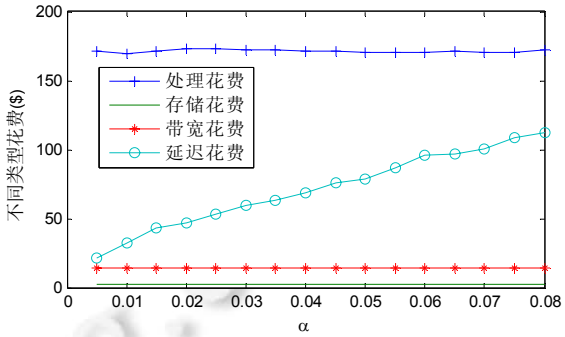

Fig.11 Variance of each type of cost with α

图 11 各类花费随参数α的变化

5.5 对比实验

本节将本文算法与其他算法进行对比,包括文献[23]中的 OLM 算法以及一些传统算法.OLM 算法为文献[2]设计的启发式策略在线算法,由于本文未考虑数据中心间迁移的费用,为公平起见,我们将 OLM 算法产生总费用减去数据中心间数据迁移费用作为比较对象.其他算法由不同的数据分配策略以及资源供给策略组合而成.在数据分配部分,主要考虑 3 种代表性策略.

1) 就近分配原则(proximity-aware data allocation,简称 PDA),将各数据源产生的数据分配至离其最近的数据中心中.显然,此策略具有最小的延迟,适合对于延迟敏感的场景;
2) 负载均衡分配原则(load balancing data allocation,简称 LBDA),将数据分配至具有最小负载的数据中心.此策略能够保持各数据中心的负载均衡;
3) 价格最低分配原则(minimal price data allocation),将数据分配至当前时刻资源价格最低的数据中心,以降低费用.

至于资源供给部分,主要考虑了两种简单策略.

1) 启发式策略(heuristic vm provisioning,简称 HVP),此策略基于历史时刻的资源需求决定当前时刻虚拟机资源供给量.为应对负载的波动性强的问题,我们在前一时刻所需要的资源量上增加 50%作为当前时刻的资源需求量;
2) 固定式策略(stable VM provisioning,简称 SVP),即,每种类型虚拟机保持固定供给量.为便于比较,我们将这一固定值设置为本文算法所得结果的平均值.显然,此策略在 T 时刻内的总量与本文算法所供给的总量相等.

另外,将以上各策略进行组合,可形成以下的不同方案:本文算法、OLM、SVP+PDA、SVP+LBDA、SVP+MPDA、HVP+PDA、HVP+LBDA、HVP+MPDA.本文在 Worldcup 数据集和 Youtube 数据集上进行了对比.

(a) Worldcup 数据集.

图 12 展现了不同方案的时间平均费用对比,仔细分析该图,可得出以下结论:

1) 除了方案 SVP+PDA 之外,本文算法比其他算法在费用上都更优.因为这种方案将数据分配至距数据源最近的数据中心进行处理,必然导致最小的延迟费用.然而,由于方案 SVP+PDA 所对应的队列负载随着时间递增而递增(如图 13 所示),意味着不能保证系统的长时间运行,因此从实际情况来讲,方案 SVP+PDA 是不可行的.又如第 5.3 节所分析,本文算法具有保持数据本地化的特性.因此,考虑到以上结果,本文算法能够在数据本地化以及系统稳定性之间进行平衡;
2) 资源供给量相同情况下(如 SVP),LBDA 数据分配策略产生了最高的花费.我们认为:这主要是因为在负载均衡的数据分配策略情况下,数据分配至各数据中心是均等的,因而不计成本地远距离迁移数据.例如,从美国迁移大规模数据至巴黎,必然导致高昂的延迟代价以及计算费用.值得注意的是,文献[23]算法 OLM 费用高于本文算法甚至简单资源供给策略 SVP.这是因为算法 OLM 未考虑资源供给策略,

对所有的数据按需及时处理(每时刻到达数据不累积至下一时刻),而其他算法则允许一定的队列延迟,因而 OLM 侧重时效性更强的应用.

结合图 7 中结论,本文算法能够通过调节参数 V 以适应不同的队列延迟,能够适应不同时效性的应用,因此相比 OLM 灵活性更强.

图 13 展示了各方案长时间运行的队列变化情况.显然,在长时间运行后,本文算法与方案 OLM,HVP+PDA,HVP+LBDA,HVP+MPDA 都能保持稳定.然而,其他策略的队列长度随着时间的增长而增长,长时间后必然导致系统的瘫痪.又注意到:SVP 资源供给策略与本文策略所供给的资源量是相同的,却比本文算法产生更高的费用以及更低的系统稳定性,因此,本文算法能够在数据分配和资源供给决策之间进行优化以降低总体费用并提高系统稳定性.如前文所述,HVP 所供给的虚拟机资源量是在前一个时隙所需的基础上增加额外 50%.尽管利用了这些策略能够保持很好的稳定性,但却以增加过高的费用为代价.而算法 OLM 平均队列负载为 0,这是因为该算法采用按需供给虚拟机资源,在每时刻都能将达到的数据处理完毕.

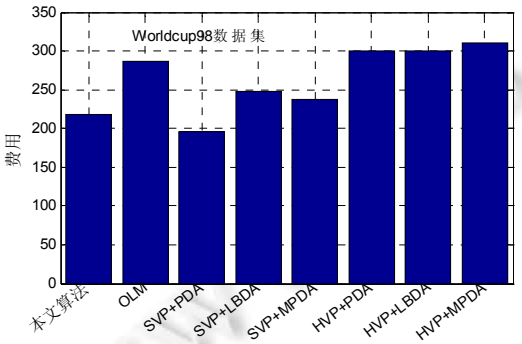

Fig.12 Comparison of cost with other strategies

图 12 与其他算法在费用上的对比

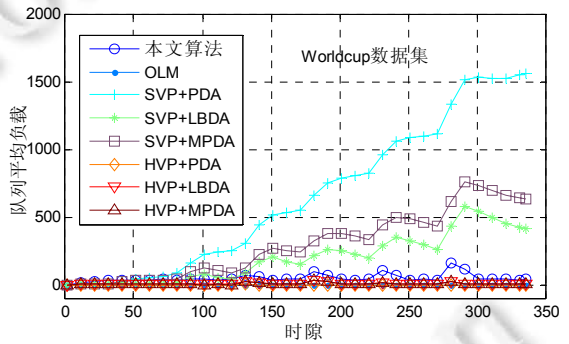

Fig.13 Comparison of queue stability with other strategies

图 13 与其他算法在队列稳定性上的对比

(b) Youtube 数据集.

图 14 和图 15 展示了不同算法在 Youtube 数据集上的对比,对比结果与数据集 Worldcup 相似,说明本文算法能够适应不同数据集,更进一步证明了算法的有效性.

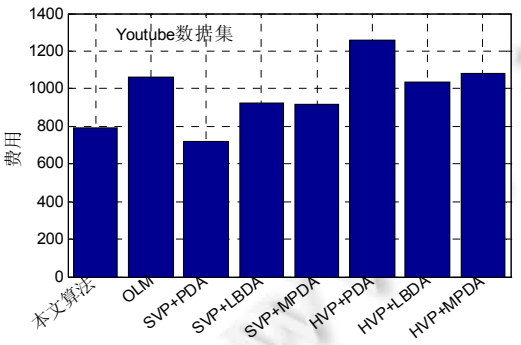

Fig.14 Comparison of cost with other strategies

图 14 与其他算法在费用上的对比

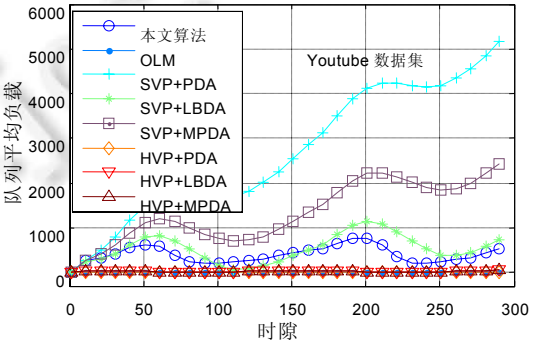

Fig.15 Comparison of queue stability with other strategies

图 15 与其他算法在队列稳定性上的对比

5.6 算法实时性

为验证算法的实时性,本文在实验中记录每次算法 CPU 运行时间.通过算法 1 可知,算法的运行时间主要与

问题规模(即数据源数 R、数据中心数 D 以及虚拟机类型数 K)有关.实验中对不同问题规模进行了记录,所得运行时间见表 2.

Table 2 Executing time of the algorithm with parameter changes

表 2 算法运行时间随参数变化

R	D	K	运行时间(ms)
4	12	5	0.56
40	12	5	0.65
4	120	5	4.68
4	12	50	0.56
40	120	50	4.92

实验结果表明,算法能在毫秒级的时间内运行完成.尽管在实际系统中情况会有所不同,但算法的在线实现是可期的.另外,通过问题规模各参数的变化情况可知,算法的复杂度主要由数据中心 D 决定,这与上文的算法复杂度分析结果相一致.

6 结 论

随着地理分散的数据源源不断产生并需要处理,采用跨区域、分布式数据中心进行大数据处理已经成为许多公司和机构关注的解决方案.如何高效地对此模式下的数据与资源进行管理成为亟待解决的问题.为此,本文设计了一种面向云端大数据处理、集数据迁移和资源供给为一体的成本最小化理论框架.通过平衡跨数据中心数据处理产生的带宽费用、存储费用、计算费用以及延迟费用等 4 种费用,该文将成本最优化问题建模成联合的随机整数优化问题.通过利用李雅普诺夫优化理论,我们将原问题转化为两个可在线解决的子问题,每个子问题又恰好对应数据分配和资源供给问题.理论分析表明:该算法能够达到较低的费用,并确保数据处理在一定的时间序列内完成.通过大量的实验,进一步验证了该算法的有效性以及相比其他典型算法的优越性.

References:

[1] 145 Amazing YouTube statistics. http://expandedramblings.com/index.php/youtube-statistics/

[2] Dewdney PE, Hall PJ, Schilizzi RT, Lazio TJLW. The square kilometre Array. Proc. of the IEEE, 2009,97(8):1482−1496. [doi: 10.1109/JPROC.2009.2021005]

[3] Dean J and Ghemawat S. MapReduce: Simplified data processing on large clusters. Communications of the ACM, 2008,51: 107−113. [doi: 10.1145/1327452.1327492]

[4] Zaharia M, Chowdhury M, Franklin MJ, Shenker S, Stoica I. Spark: Cluster computing with working sets. In: Proc. of the 2nd USENIX Conf. on Hot Topics in Cloud Computing (HotCloud 2010). 2010.

[5] Melnik S, Gubarev A, Long JJ, Romer G. Dremel: Interactive analysis of Web-scale datasets. Proc. of the VLDB Endowmenet, 2010,3(1-2):330−339. [doi: 10.14778/1920841.1920886]

[6] Pregel. http://kowshik.github.io/JPregel

[7] Yin J, Wang ZS, Li Q, Su WJ. Personalized recommendation based on large-scale implicit feedback. Ruan Jian Xue Bao/Journal of Software, 2014,25(9):1953−1966 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/4648.html [doi: 10.13328/j.cnki.jos.004648]

[8] Han L, Li M. Open source software classification using cost-sensitive multi-label learning. Ruan Jian Xue Bao/Journal of Software, 2014,25(9):1982−1991 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/4639.html [doi: 10.13328/j.cnki.jos.004639]

[9] Xie JY, Gao HC. Statistical correlation and K-means based distinguishable gene subset selection algorithms. Ruan Jian Xue Bao/Journal of Software, 2014,25(9):2050−2075 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/4644.html [doi: 10.13328/j.cnki.jos.004644]

[10] Vlachou A, Doulkeridis C, Nørvåg K, Vazirgiannis M. On efficient top-k query processing in highly distributed environments. Distributed and Parallel Databases, 2012,30(3-4):239−271. [doi: 10.1007/s10619-012-7094-2]

[11] Xu J, Wang GY, Yu H. Review of big data processing based on granular computing. Chinese Journal of Computers, 2015,38(8): 1497−1517 (in Chinese with English abstract).

[12] Ding YW, Qin XL, Liu L, Wang TC. An energy efficient algorithm for big data processing in heterogeneous cluster. Journal of Computer Research and Development, 2015,52(2):377−390 (in Chinese with English abstract).

[13] Li W, Zhang DF, Huang K, Xie K. Accurate multi-dimension counting bloom filter for big data processing. Acta Electronica Sinica, 2015,43(4):752−657 (in Chinese with English abstract).

[14] Lu ZM, Feng JG, Fan DM, Yang P, Tian Y. Novel partitional clustering algorithm for large data processing. System Engineering and Electronics, 2014,36(5):1010−1015 (in Chinese with English abstract).

[15] Moving an elephant: Large scalehadoop data migration at Facebook. 2016. http://www.facebook.com/notes/paul-yang/moving-an-elephant-large-scale-hadoop-data-migration-at-facebook/10150246275318920

[16] Schadt EE, Linderman MD, Sorenson J, Lee L, Nolan GP. Computational solutions to large-scale data management and analysis. Nat Rev Genet, 2010,11:647−657. [doi: 10.1038/nrg2857.]

[17] Facebook's prism project. 2016. http://www.wired.com/wiredenterprise/2012/08/facebook-prism/

[18] Corbett JC, Dean J, Epstein M, *et al*. Spanner: Google's globally-distributed database. In: Proc of the OSDI 2012. 2012.

[19] Connecting Geographically Dispersed Data Centers. HP FlexFabric Interconnect, Fact Sheet, 2015.

[20] Interconnecting Geographically Dispersed Data Centers Using VPLS-Design and System Assurance Guide. Cisco Systems, Inc., 2009.

[21] Cho B, Gupta I. Budget-constrained bulk data transfer via internet and shipping networks. In: Proc. of the ACM ICAC. 2011. 71−80. [doi: 10.1145/1998582.1998595]

[22] Cho B, Gupta I. New algorithms for planning bulk transfer via internet and shipping networks. In: Proc. of the IEEE ICDCS. IEEE, 2010. 305−314. [doi: 10.1109/ICDCS.2010.59]

[23] Zhang LQ, Wu C, Li ZP, Guo CX, Chen MH, Lau Francis CM. Moving big data to the cloud: An online cost-minimizing approach. IEEE Journal on Selected Areas in Communications, 2013,31(12):2710−2721. [doi: 10.1109/JSAC.2013.131211]

[24] Zhang LQ, Li ZP, Wu C, Chen MH. Online algorithms for uploading deferrable big data to the cloud. In: Proc. of the IEEE INFOCOM. 2014. 2022−2030. [doi: 10.1109/INFOCOM.2014.6848143]

[25] Liu L, Wang H, Liu X, Jin X, He WB, Wang QB, Chen Y. GreenCloud: A new architecture for green data center. In: Proc. of the 6th Int'l Conf. on Industry Session on Autonomic Computing and Communications Industry Session. 2009. [doi: 10.1145/1555312.1555319]

[26] Petrucci V, Loques O, Mosse D. A dynamic configuration model for power-efficient virtualized server clusters. In: Proc. of the 11th Brazillian Workshop on Real-Time and Embedded Systems. 2009.

[27] Tang JH, Tay WP, Wen YG. Dynamic request redirection and elastic service scaling in cloud-centric media networks. IEEE Trans. on Multimedia, 2014,16(5):1434−1445. [doi: 10.1109/TMM.2014.2308726]

[28] Tassiulas L, Ephremides A. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. IEEE Trans. on Automatic Control, 1992,37(12):1936−1948. [doi: 10.1109/9.182479]

[29] Urgaonkar R, Kozat U, Igarashi K, Neely MG. Dynamic resource allocation and power management in virtualized data centers. In: Proc. of the IEEE Network Operations and Management Symp. (NOMS 2010). 2010. 479−486. [doi: 10.1109/NOMS.2010.5488484]

[30] Zhou Z, Liu FM, Xu Y, Zou R, Xu H, Lui JCS, Jin H. Carbon-Aware load balancing for geo-distributed cloud services. In: Proc. of the IEEE 21st Int'l Symp. on Modelling, Analysis and Simulation of Computer and Telecommunication Systems. 2013. 232−241. [doi: 10.1109/MASCOTS. 2013.31]

[31] Wu D, Xue Z, He J. iCloudAccess: Cost-Effective streaming of video games from the cloud with low latency. IEEE Trans. on Circuits and Systems for Video Technology, 2014,24(8):1405−1416. [doi: 10.1109/TCSVT.2014.2302543]

[32] Zhou Z, Liu FM, Jin H, Li B, Li BC, Jiang HB. On arbitrating the power-performance tradeoff in SaaS clouds. In: Proc. of the IEEE INFOCOM. 2013. [doi: 10.1109/INFCOM.2013.6566875]

[33] Niu YP, Luo B, Liu FM, Liu JC, Li B. When hybirid cloud meeets flash crowd: Towards cost-effective service provisioning. In: Proc. of the IEEE INFOCOM. 2015. [doi: 10.1109/INFOCOM.2015.7218477]

[34] Yao Y, Huang LB, Sharma A, Golubchik L, Neely MJ. Power cost reduction in distributed data centers: A two-time-scale approach for delay tolerant workloads. IEEE Trans. on Parallel and Distributed Systems, 2014,25(1):200−211. [doi: 10.1109/TPDS.2012.341]

[35] Zhao J, Li HX, Wu C, Li ZP, Zhang ZZ, Lau Francis CM. Dynamic pricing and profit maximization for clouds with geo-distributed datacenters. In: Proc. of the IEEE INFOCOM. 2014. [doi: 10.1109/INFOCOM.2014.6847931]

[36] Neely MJ. Stochastic Network Optimization with Application to Communication and Queueing Systems. Morgan and Claypool, 2010.

[37] Neely MJ. Opportunistic scheduling with worst case delay guarantees in single and multi-hop networks. In: Proc. of the INFOCOM. IEEE, 2011. 1728−1736. [doi: 10.1109/INFCOM.2011.5934971]

[38] Arlitt M, Jin T. A workload characterization study of the 1998 world cup Web site. IEEE Network, 2000,14(3):30−37. [doi: 10.1109/65.844498]

[39] Umasstracerepository. 2014. http://traces.cs.umass.edu/index.php/Network/Network

[40] Where Amazon's data centers are located. http://www.datacenterknowledge.com/archives/2008/11/18/where-amazons-data-centers-are-located/

[41] Gpsspg. http://www.gpsspg.com/distance.htm

附中文参考文献:

[7] 印鉴,王智圣,李琪,苏伟杰.基于大规模隐式反馈的个性化推荐.软件学报,2014,25(9):1953−1966. http://www.jos.org.cn/1000-9825/4648.html [doi: 10.13328/j.cnki.jos.004648]

[8] 韩乐,黎铭.基于代价敏感多标记学习的开源软件分类.软件学报,2014,25(9):1982−1991. http://www.jos.org.cn/1000-9825/4639.html [doi: 10.13328/j.cnki.jos.004639]

[9] 谢娟英,高红超.基于统计相关性与 *K*-means 的区分基因子集选择算法.软件学报,2014,25(9):2050−2075. http://www.jos.org.cn/1000-9825/4644.html [doi: 10.13328/j.cnki.jos.004644]

[11] 徐计,王国胤,于洪.基于粒计算的大数据处理.计算机学报,2015,38(8):1497−1517.

[12] 丁有伟,秦小麟,刘亮,王涛春.一种异构集群中能量高效的大数据处理算法.计算机研究与发展,2015,52(2):377−390.

[13] 李玮,张大方,黄昆,谢鲲.面向大数据处理的高精度多维计数布鲁姆过滤器.电子学报,2015,43(4):752−657.

[14] 卢志茂,冯进玫,范冬梅,杨朋,田野.面向大数据处理的划分聚类新方法.系统工程与电子技术,2014,36(5):1010−1015.

附录 A:漂移惩罚上界推导

证明:根据$(\max[x-y,0]+z)^2\leqslant x^2+y^2+z^2+x(z-y)$,对向量$\boldsymbol{\Theta}(t)=[\boldsymbol{Z}(t),\boldsymbol{H}(t)]$则有:

$$\begin{aligned}Z_d(t+1)^2-Z_d(t)^2\leqslant&\left\{1_{(H(t)>0)}\left(\varepsilon_d-\sum_{k\in\mathcal{K}}n_d^k(t)v_k\right)-1_{(H(t)=0)}\sum_{k\in\mathcal{K}}n_d^{k,\max}v_k\right\}+\\&2Z_d(t)\left\{1_{(H(t)>0)}\left(\varepsilon_d-\sum_{k\in\mathcal{K}}n_d^k(t)v_k\right)-1_{(H(t)=0)}\sum_{k\in\mathcal{K}}n_d^{k,\max}v_k\right\}\\\leqslant&(\varepsilon_d)^2+\left(\sum_{k\in\mathcal{K}}n_d^{k,\max}v_k\right)^2+2Z_d(t)\left(\varepsilon_d-\sum_{k\in\mathcal{K}}n_d^k(t)v_k\right)\end{aligned}\tag{A.1}$$

$$H_d(t+1)^2-H_d(t)^2\leqslant\left(\sum_{k\in\mathcal{K}}n_d^k(t)v_k\right)^2+\left(\sum_{r\in\mathcal{R}}\lambda_r^d(t)\right)^2+2H_d(t)\left(\sum_{r\in\mathcal{R}}\lambda_r^d(t)-\sum_{k\in\mathcal{K}}n_d^k(t)v_k\right)\tag{A.2}$$

由于 $\lambda_r^d(t)$ 和 $n_d^k(t)$ 的最大值分别为 $A_{\max}^r$ 和 $N_d^{k,\max}$,设 $B=\frac{1}{2}\sum_{d\in\mathcal{D}}\left\{2\left(\sum_{k\in\mathcal{K}}N_d^{k,\max}v_k\right)^2+(\varepsilon_d)^2+\left(\sum_{r\in\mathcal{R}}A_{\max}^r\right)\right\}$,则可得单时隙李雅普诺夫漂移如下:

$$\begin{aligned}\Delta(\boldsymbol{\Theta}(t))=&\frac{1}{2}\sum_{d\in\mathcal{D}}\mathbb{E}\{H_d(t+1)^2-H_d(t)^2\mid\boldsymbol{\Theta}(t)\}+\frac{1}{2}\sum_{d\in\mathcal{D}}\mathbb{E}\{Z_d(t+1)^2-Z_d(t)^2\mid\boldsymbol{\Theta}(t)\}\\\leqslant&B+\sum_{d\in\mathcal{D}}\mathbb{E}\left\{H_d(t)\left(\sum_{r\in\mathcal{R}}\lambda_r^d(t)-\sum_{k\in\mathcal{K}}n_d^k(t)v_k\right)\middle|\boldsymbol{\Theta}(t)\right\}+\sum_{d\in\mathcal{D}}\mathbb{E}\left\{Z_d(t)\left(\varepsilon_d-\sum_{k\in\mathcal{K}}n_d^k(t)v_k\right)\middle|\boldsymbol{\Theta}(t)\right\}\end{aligned}\tag{A.3}$$

通过在上述表达中加上 $V\cdot\mathbb{E}\{C(t)\}$,我们得到了公式(20)中的漂移惩罚界限. □

附录 B:定理 1 的证明

为证明定理 1,我们首先给出如下引理.

引理 B.1(最优随机稳定策略的存在性). 至少存在一个策略π,使其选择的可行解 $n_d^{k,\pi}(t),\lambda_d^{k,\pi}(t)$ ($\forall d\in\mathcal{D}$, $r\in\mathcal{R},k\in\mathcal{K},t\in[1,T]$)满足:

$$\begin{aligned}&\mathbb{E}\{C(t)\}=C^*\\&\mathbb{E}\left\{\sum_{r\in\mathcal{R}}\lambda_r^{d,\pi}(t)\right\}\leqslant\mathbb{E}\left\{\sum_{k\in\mathcal{K}}n_d^{k,\pi}(t)v_k\right\}\\&\varepsilon_d\leqslant\mathbb{E}\left\{\sum_{k\in\mathcal{K}}n_d^{k,\pi}(t)v_k\right\}\end{aligned}\tag{A.4}$$

其中,C^*是理论成本下界.该引理可利用卡拉特欧多定理证明,详细见文献[36].

根据引理 B.1,我们可如下证明公式(27).

证明:从引理 B.1 可知,必然存在一个常数δ>0,满足:

$$\mathbb{E}\left\{\sum_{r\in\mathcal{R}}\lambda_r^{d,\pi}(t)\right\}\leqslant\mathbb{E}\left\{\sum_{k\in\mathcal{K}}n_d^{k,\pi}(t)v_k\right\}-\delta\tag{A.5}$$

以及

$$\varepsilon_d\leqslant\mathbb{E}\left\{\sum_{k\in\mathcal{K}}n_d^{k,\pi}(t)v_k\right\}-\delta\tag{A.6}$$

由本文算法力求最小化不平等公式(20)的右半部分,通过在各时隙所有可行的决策中选择最优决策,应用引理 B.1,将公式(A.5)和公式(A.6)代入公式(20),我们可得:

$$\Delta(\Theta(t))+V\cdot\mathbb{E}\{C(t)\,|\,\Theta(t)\}\leqslant B+VC^*-\delta\sum_{d\in\mathcal{D}}\mathbb{E}\{H_d(t)\}-\delta\sum_{d\in\mathcal{D}}\mathbb{E}\{Z_d(t)\}\tag{A.7}$$

取公式(A.7)的期望,并应用$\Delta(\Theta(t))=\mathbb{E}\{L(\Theta(t+1))-L(\Theta(t))|\Theta(t)\}$,可得:

$$\mathbb{E}\{L(\Theta(t+1))-L(\Theta(t))\,|\,\Theta(t)\}+V\cdot\mathbb{E}\{C(t)\,|\,\Theta(t)\}\leqslant B+VC^*-\delta\sum_{d\in\mathcal{D}}\mathbb{E}\{H_d(t)\}-\delta\sum_{d\in\mathcal{D}}\mathbb{E}\{Z_d(t)\}\tag{A.8}$$

在时隙 t=1,…,T–1 上应用伸缩和定理(telescoping sum law),并将结果除以 T,可得:

$$\frac{\mathbb{E}\{L(\Theta(T))-L(\Theta(0))\}}{T}+\frac{V}{T}\cdot\sum_{t=0}^{T-1}\mathbb{E}\{C(t)\,|\,\Theta(t)\}\leqslant B+VC^*-\frac{\delta}{T}\sum_{t=0}^{T-1}\sum_{d\in\mathcal{D}}\mathbb{E}\{H_d(t)\}-\frac{\delta}{T}\sum_{t=0}^{T-1}\sum_{d\in\mathcal{D}}\mathbb{E}\{Z_d(t)\}\tag{A.9}$$

对上述公式进行整理,并考虑约束:$L(\Theta(0))=0,H_d(t)\geqslant0,Z_d(t)\geqslant0$,得:

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\{C(t)\}\leqslant C^*+\frac{B}{V}\tag{A.10}$$

对公式(A.10)求极限 $T\to\sim$,可得公式(26). □

附录 C:定理 2 的证明

证明:对于 $Z_d(t)$,已知 $Z_d(0)=0<Z_d^{\max}$.对任意时刻 $t\in[0,T]$:

- 若 $Z_d(t)\leqslant\dfrac{Vp_d^{\max}}{v_{\min}}$,则 $Z_d(t+1)\leqslant\dfrac{Vp_d^{\max}}{v_{\min}}+\varepsilon_d=Z_d^{\max}$ (因为如公式(14)所示,$Z_d(t)$最大增量为ε_d);
- 若 $Z_d(t)>\dfrac{Vp_d^{\max}}{v_{\min}}$,则 $H_d(t)+Z_d(t)>\dfrac{Vp_d^{\max}}{v_{\min}}>\dfrac{Vp_d^k}{v_k}$,并且队列将减少 $\sum_{k\in\mathcal{K}}N_d^{k,\max}v_k$ (见公式(26)).

而 $\varepsilon_d<\sum_{k\in\mathcal{K}}N_d^{k,\max}v_k$,此时队列 $Z_d(t)$的增量小于减小量,因此 $Z_d(t+1)<Z_d(t)<Z_d^{\max}$.队列 $Z_d(t)$的界限得证.

类似地,对于队列 $H_d(t)$, $H_d(0)=0<H_d^{\max}$.对任意时刻 t:

- 若 $H_d(t)\leqslant\frac{Vp_d^{\max}}{v_{\min}}$,则 $H_d(t+1)\leqslant\frac{Vp_d^{\max}}{v_{\min}}+\sum_{r\in\mathcal{R}}A_{\max}^r$ (由公式(13),$H_d(t+1)$最大增量为 $\sum_{r\in\mathcal{R}}A_{\max}^r$);
- 若 $H_d(t)>\frac{Vp_d^{\max}}{v_{\min}}$,则 $H_d(t)+Z_d(t)>\frac{Vp_d^{\max}}{v_{\min}}\geqslant\frac{Vp_d^k}{v_k}$,由公式(26)可知,$t+1$ 时刻队列将减小 $\sum_{k\in\mathcal{K}}N_d^{k,\max}v_k$.

而 $\sum_{r\in\mathcal{R}}A_{\max}^r\leqslant\sum_{k\in\mathcal{K}}N_d^{k,\max}v_k$,即队列的增量小于减小量,因而 $H_d(t+1)<H_d(t)<H_d^{\max}$. □

因此,队列 $H_d(t)$的界限得证.

附录 D:定理 3 的证明

证明:若存在$\tau\in[t+1,t+l]$满足 $H_d(\tau)=0$,则在 t 时刻到达的数据将在 l 个时间序列内被处理.若 $H_d(\tau)>0,\tau\in[t+1,t+l]$,根据公式(14),可得:

$$Z_d(\tau+1)=\max\left[Z_d(\tau)+\varepsilon_d-\sum_{k\in\mathcal{K}}n_d^k(\tau)v_k,0\right]\geqslant Z_d(\tau)+\varepsilon_d-\sum_{k\in\mathcal{K}}n_d^k(\tau)v_k \tag{A.11}$$

在时间序列$\tau\in[t+1,t+l]$内对以上的不等式求和,可得:

$$Z_d(t+l+1)\geqslant Z_d(t+1)+l\cdot\varepsilon_d-\sum_{\tau=t+1}^{t+l+1}\sum_{k\in\mathcal{K}}n_d^k(\tau)\cdot v_k \tag{A.12}$$

因此,可以推导出 $\sum_{\tau=t+1}^{t+l+1}\sum_{k\in\mathcal{K}}n_d^k(\tau)\cdot v_k\geqslant l\cdot\varepsilon_d-Z_d^{\max}$. 注意:当 $l=[H_d^{\max}+Z_d^{\max}/\varepsilon_d]$ 时,有 $l\cdot\varepsilon_d-Z_d^{\max}=H_d^{\max}$, 则

$$\sum_{\tau=t+1}^{t+l+1}\sum_{k\in\mathcal{K}}n_d^k(\tau)\cdot v_k\geqslant H_d^{\max}\geqslant H_d(t) \tag{A.13}$$

由于队列系统以先进先出方式运行,在 t 时刻到达的数据将比在 t 时刻之后到达的数据先处理.既然 l 个时间序列被处理的总数据量大于 $H_d(t)$,那么所有在 t 时刻到达的数据都将在 l 个时间序列中被处理,即最差的延迟为 l 个时隙. □

肖文华(1988－),男,湖南桂阳人,博士生,主要研究领域为云计算,移动云计算中的资源管理问题.

包卫东(1971－),男,博士,教授,博士生导师,主要研究领域为信息系统.

朱晓敏(1979－),男,博士,副教授,CCF 高级会员,主要研究领域为云计算资源管理,实时系统.

邵屹杨(1992－),男,硕士生,主要研究领域为云计算容错管理研究.

陈超(1990－),男,博士生,主要研究领域为云计算,移动云计算.

Jianhong WU (1964－),男,博士,教授,博士生导师,主要研究领域为微分方程动力学,面向大数据分析的神经网络架构.