DOI：10.7652/xjtuxb201407014

云计算环境下的动态反馈作业调度算法

马莉[1]，唐善成[1]，王静[1]，赵安新[2]
(1.西安科技大学通信与信息工程学院，710054，西安；2.西安交通大学电气工程学院，710049，西安)

摘要：针对现有Hadoop作业调度算法在多用户、异构环境下不具备反馈机制的问题，提出一种云计算环境下具备反馈机制的动态作业调度算法。该算法引入排队论模型，采用单队列多资源池服务窗口的设计思路，将所有作业统一提交到一个支持优先级的排队队列，作业分发控制模块选择优先级最高的作业分发到空闲的资源池窗口执行；Hadoop集群通过自身的心跳机制将作业运行的初始化时间、运行时间等信息传递给参数统计模块进行统计，将获得的平均到达率和平均服务率这两个核心参数的实际值传递给反馈机制模块，根据调度算法模型计算出平均逗留时间和平均队长的理论值并与实际值进行对比，当差值大于阈值时对该调度算法的核心参数进行适当调节使差值收敛于阈值，将具有较大平均逗留时间和平均队长的作业调度到有槽位数的资源池服务窗口执行。实验结果表明：与经典算法相比，该算法具有较高的调度效率和负载平衡能力，作业平均逗留时间比先进先出调度算法和公平调度算法分别减少了57%和19%，平均队长分别减少了50%和37%。

关键词：云计算；Hadoop集群；作业调度；排队论；反馈机制
中图分类号：TP311 **文献标志码**：A **文章编号**：0253-987X(2014)07-0077-06

Dynamic Feedback Job Scheduling Algorithm in Cloud Computing

MA Li[1], TANG Shancheng[2], WANG Jing[1], ZHAO Anxin[3]
(1. School of Communication and Information, Xi'an University of Science and Technology, Xi'an 710054, China;
2. School of Electrical Engineering, Xi'an Jiaotong University, Xi'an 710049, China;)

Abstract: Aiming at the defectiveness in the feedback mechanism of current Hadoop scheduling algorithms in a multi-user and heterogeneous environment, a dynamic job scheduling algorithm with feedback mechanism is proposed, where a kind of queues theory model is introduced, a single queue design and multiple resource pool service windows are adopted. All jobs are submitted to a single queue which supports priority, and then the job with the highest priority is scheduled to the free resource pool window to run by the job distributing and controlling module. With its own heartbeat mechanism of Hadoop cluster, the job-related running data, such as initializing time and running time, are transmitted to the parametric statistics module to get the actual mean arrival rate and service rate to transfer to the feedback mechanism module. The theoretical values of mean staying time and queue-length can be calculated by this job scheduling model and compared with the actual values. If the difference is larger than the threshold value, the core parameters are suitably adjusted to make the difference converging to the threshold value, which can schedule the jobs with larger mean staying time and queue-length to the resource pool service window with slots. Simulations demonstrate that this scheduling algorithm is advantageous in scheduling efficiency and load balancing compared with those classic ones. The

收稿日期：2014-01-03。 作者简介：马莉(1979—)，女，博士生，讲师。 基金项目：国家自然科学基金资助项目(61302133)。
网络出版时间：2014-05-29 网络出版地址：http://www.cnki.net/kcms/detail/61.1069.T.20140529.1207.001.html

mean staying time is shortened by 57% and 19% compared with FIFO and FAIR scheduling algorithms, and the mean queue-length is reduced by 50% and 37%, respectively.

Keywords: cloud computing; Hadoop clusters; job scheduling; queues theory; feedback mechanism

随着高性能计算集群规模的不断扩充,集群系统出现了利用率低、能耗高等问题,而作业调度系统的应用取代了人工调度,显著提高了高性能计算集群系统的利用率,实现节能降耗。传统的调度方法有顺序调度、随机分配、轮询调度等,这些方法虽然易于实现,但在云计算环境下,尤其是异构环境下,由于集群中不同节点的计算能力不同,传统的调度方法往往使集群的处理性能不佳,因此作业调度成为云计算的一个研究热点。

云计算作业调度的目标是在满足用户 QoS 的同时,实现最优跨度,并尽可能确保系统负载均衡和经济性[1-2]。但是,云计算平台的异构性和动态性使云环境下的作业调度研究变得复杂和困难[3],因此如何提高云计算环境下作业调度的公平性和实现效率就成为一个重要的研究课题。Google 提出的 MapReduce 编程模型是一种高效的任务并行计算模型,用于 PB 级大规模数据处理,目前像 Amazon[4]、IBM 蓝云[5]、Yahoo、微软、百度等云平台都在其基础架构上采用了 MapReduce 模型的设计思想。

Hadoop 作为 MapReduce 计算模型的开放式源码框架,是目前研究及应用最为广泛的开源云计算软件平台[6]。Hadoop 作业调度算法主要针对集群资源利用率低及用户作业运行时间长等问题,然而现有的先进先出调度器(first in first out,FIFO)、公平调度器(FAIR)、计算能力调度器(Capacity)和 LATE(Longest Approximate Time to End)这 4 种 Hadoop 作业调度器算法各有优劣。Hadoop 默认采用 FIFO 这种先来先服务的调度模式,即按照作业提交的顺序将作业顺序排队执行,这种单用户调度算法缺乏相应的灵活性[7]。为解决作业调度的公平性并提高计算能力,雅虎和 Facebook 开发了 FAIR 调度器[8]和 Capacity 调度器[9],虽然这两种调度器算法支持多用户和多队列,但都只适合于同构环境下。针对异构环境下的 LATE 调度器[10]虽然采用了备份任务的方法来解决异构环境下的作业调度,但是备份任务越多系统资源浪费就越大,备份任务执行和正常作业执行还会竞争系统资源。同时,这几种调度算法均不具备反馈机制。

因此,本文将排队论的 $M/M/S/\infty$ 模型[11]引入到云作业调度领域,基于单队列多资源池服务窗口的设计思路,提出一种适合于多用户、异构环境的具备反馈机制的动态云作业调度算法(MMS)。在大作业量的情况下,MMS 调度算法可以有效缩短 Hadoop 集群作业队列的平均逗留时间和平均队长,提高 Hadoop 集群的作业调度效率和负载均衡能力。

1 云作业调度的 $M_1/M_2/S/\infty$ 排队模型

排队论是研究系统随机聚散现象和随机服务系统工作过程的数学理论和方法,又称随机服务系统理论,针对电话通话问题运用概率论方法来开展研究,目的是对各个服务系统进行正确设计,使之发挥最佳效益。在 $M/M/S/\infty$ 排队模型[11]中,第 1 个符号表示顾客相继到达时间服从负指数分布,第 2 个符号表示服务时间服从负指数分布,第 3 个符号表示系统内有 S 个服务窗口,第 4 个符号表示系统内顾客最大排队容量为 ∞。本文将 $M/M/S/\infty$ 排队模型引入云作业调度系统,提出一种 $M_1/M_2/S/\infty$ 云作业调度模型。其中:M_1 表示作业相继到达的时间服从参数为 γ 的负指数分布,γ 为单位时间内平均到达的作业数(平均到达率);M_2 表示服务时间(即作业执行时间)的分布服从参数为 μ 的负指数分布,μ 为单位时间内平均服务完成的作业数(平均服务率);S 表示资源池服务窗口的数量;∞ 表明系统容量无限。

如果把 Hadoop 云计算平台中的作业调度看作一个随机服务系统,则可将随机到达的用户作业看作为排队论模型中的客户,队列就是调度系统中的资源池队列,服务机构就是 Hadoop 集群中的计算槽位,因此按照排队论模型,Hadoop 默认的 FIFO 就是一个单资源池服务窗口 $M_1/M_2/1/\infty$ 模型的排队系统。

与 FIFO 调度类似,FAIR 调度和 Capacity 调度都支持多队列,但是作业一旦被提交到某个队列,该作业就只能在当前队列内调度。因此,FAIR 调度和 Capacity 调度都属于 S 个 $M_1/M_2/1/\infty$ 模型,而 S 个 $M_1/M_2/1/\infty$ 模型的效率低于 $M_1/M_2/S/\infty$

模型(2.2 节将通过理论推导予以证明)。

2　MMS 云作业调度模型理论推导

2.1　MMS 云作业调度模型的简化推导

MMS 模型(即 $M_1/M_2/S/\infty$模型)最简单的情况就是当 $S=1$，这时 $M_1/M_2/S/\infty$模型就简化为 $M_1/M_2/1/\infty$模型，即单资源池服务窗口 FIFO 调度。

2.1.1　FIFO 调度的作业队长分布　系统的状态指系统中的作业数量，系统中有 n 个作业就表明系统状态为 n。令 ρ 是集群服务强度，即正在接受服务的作业平均数，ρ 反映了 Hadoop 集群繁忙的程度，ρ 越大，表明系统资源利用率越高。记 $\rho=\gamma/\mu$，其中 γ 为平均到达率，μ 为平均服务率，设 $\rho<1$(否则队列将排至无限远)。经过推导，得出整个集群处于空闲状态的概率为

$$P_0=\left(\sum_{n=0}^{\infty}\rho^n\right)^{-1}=\left(\frac{1}{1-\rho}\right)^{-1} \tag{1}$$

记 $P_n=P\{n=N\}(n=0,1,2,\cdots)$，$P_n$ 为系统达到平衡状态后队长 N 的概率分布，对个数为 S 的多资源池服务窗口系统有 $\gamma_n=\gamma,n=0,1,2,\cdots$，经过推导得出作业队长分布为

$$P_n=(1-\rho)\rho^n,\quad n=0,1,2,\cdots \tag{2}$$

2.1.2　FIFO 调度的参数分析　根据平稳状态下 FIFO 调度的队长分布，可以计算出平均队长为

$$L_w=\sum_{n=0}^{\infty}nP_n=\frac{\rho}{1-\rho}=\frac{\gamma}{\mu-\gamma} \tag{3}$$

式中：L_w 为系统内作业排队总长度的均值，即排队等待的作业数加上正在被执行的作业数。

平均排队队长为

$$L_q=\sum_{n=1}^{\infty}(n-1)P_n=\frac{\gamma^2}{\mu(\mu-\gamma)} \tag{4}$$

式中：L_q 表示系统内排队等待执行作业数的均值。

在 Hadoop 作业系统中用户作业的逗留时间服从$(\mu-\gamma)$的负指数分布，经过推导，作业在 Hadoop 系统中的平均逗留时间为

$$T_w=\frac{1}{\mu-\gamma} \tag{5}$$

用户作业在 Hadoop 集群中的逗留时间为平均等待时间 T_q 与接受集群服务时间之和，经过推导，作业平均等待时间为

$$T_q=T_w-\frac{1}{\mu}=\frac{\gamma}{\mu(\mu-\gamma)} \tag{6}$$

从式(3)、(5)可以推导出平均队长为

$$L_w=\gamma T_w \tag{7}$$

从式(4)、式(6)可以推导出平均排队队长为

$$L_q=\gamma T_q \tag{8}$$

2.2　MMS 云作业调度模型的参数分析

本节将证明 $M_1/M_2/S/\infty$模型比 $M_1/M_2/1/\infty$模型具有更小的平均队长。对于 $M_1/M_2/S/\infty$模型，假定用户将作业单个依次提交，相继提交时间间隔服从参数为 γ 的负指数分布，系统中共有 S 个资源池服务服务窗口，每个服务窗口相互独立且服从参数为 γ 的负指数分布。当作业提交到队列后，若 S 个资源服务窗口有空闲槽位则立即调度执行，否则继续在队列中等待调度。

记 $\rho_S=\frac{\rho}{S}=\frac{\gamma}{S\mu}$，$\rho_S$ 为 S 个资源服务窗口时的集群服务强度，当 $\rho_S<1$ 时

$$P_n=\begin{cases}\dfrac{\rho^n}{n!}P_0, & n=1,2,\cdots,S\\ \dfrac{\rho^n}{S!S^{n-S}}P_0, & n\geqslant S\end{cases} \tag{9}$$

其中整个集群处于空闲状态的概率

$$P_0=\left[\sum_{n=0}^{S-1}\frac{\rho^n}{n!}+\frac{\rho^S}{S!(1-\rho_S)}\right]^{-1} \tag{10}$$

根据式(9)和式(10)，作业到达集群系统时需要等待的概率为

$$P(S,\rho)=\sum_{N=S}^{\infty}P_n=\frac{\rho^S}{S!(1-\rho_S)}P_0 \tag{11}$$

根据式(9)和式(11)得到平均排队队长为

$$L_q=\frac{P_0\rho^S\rho_S}{S!(1-\rho_S)^2} \tag{12}$$

记集群系统正在接受服务的用户作业的平均数目为 $\bar{S}$，即处于忙碌状态的资源池服务窗口平均数为

$$\bar{S}=\sum_{n=0}^{S-1}\frac{n\rho^n}{n!}P_0+S\frac{\rho^S}{S!(1-\rho_S)}P_0 \tag{13}$$

从式(13)可以得到

$$L_w=L_q+\rho \tag{14}$$

从式(14)中可以看出，平均队长只和平均排队队长有关。从式(12)中可以看到，S 越大，则平均排队队长越短。因此，当 S 最小为 1 时(即只有 1 个资源服务窗口)，MMS 即为单服务台的 FIFO 调度，此时平均排队队长的值最大，说明 FIFO 的平均队长最长；当 $S>1$ 时，此时 MMS 为多资源服务窗口调度，S 值越大，MMS 的平均排队队长就越短，相应的平均队长也越短，说明系统整体调度效率越高。

2.3　计算实例验证

假设在一个 Hadoop 集群的调度系统中有 3 个作业资源池服务窗口和 1 个排队队列，用户作业到达集群符合泊松分布，平均到达率为 $\gamma=0.4$/min，

作业执行时间服从负指数分布，平均服务率为 $\mu=0.15/\text{min}$。使用上述推理公式，得到 MMS 与 FAIR & Capacity 调度算法的性能对比数据，如表 1 所示。

表 1 MMS 与 FAIR & Capacity 调度算法的性能参数

模型	P_0	L_w	$P(S,\rho)$	T_q
MMS	0.028	9.07	0.800	16.00
FAIR & Capacity	0.133	23.55	0.867	52.16

由表 1 可见，本文提出的云作业调度 MMS 模型相对于 FAIR & Capacity 的 S 个 $M_1/M_2/1/\infty$ 模型而言，在空闲状态概率、平均队长、作业等待概率和平均等待时间等调度性能方面均有一定的优越性。

3 MMS 算法设计方案

3.1 整体算法设计

MMS 整体算法设计方案如图 1 所示。

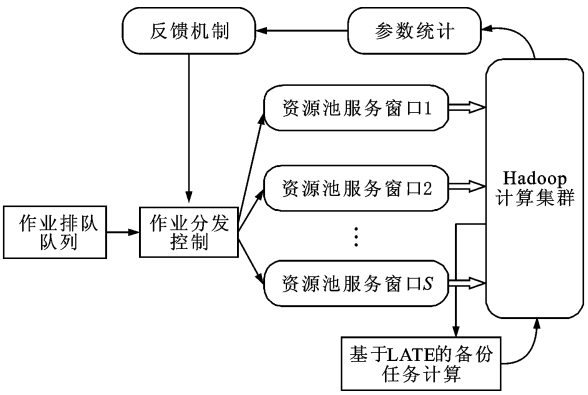

图 1 MMS 调度算法设计图

定义 1 作业排队队列。本算法模型只设计了一个排队队列，当有用户提交作业时不用指定队列，但是需要指定资源池窗口。

定义 2 资源池窗口。本算法模型中的资源池窗口类似于 FAIR & Capacity 调度算法中的队列，不同之处在于在 MMS 调度模型中的资源池窗口是作业执行窗口，里面的作业都处于运行状态，而 FAIR & Capacity 调度算法中的队列是用户根据参数指定的队列名将作业提交到相应的队列进行排队，然后在各个队列中使用一定的策略调度执行。资源池窗口用户可以在配置文件中进行配置，包括资源池名、资源池数目、每个资源池同时执行的作业数目。

整体的 MMS 调度算法流程如下：

(1)Init()；//初始化

(2)If（如果有用户提交作业）do

将作业统一提交到作业排队队列；

(3) While(作业队列不空){

在一个优先级窗口内对作业按照优先级排序；

选择一个优先级最高的作业 maxPriorityJob；

(4) 将作业 maxPriorityJob 提交到用户指定的资源池队列 Queue 中：

If(Queue 中的运行作业数目＜配置值)do

在资源池队列 Queue 中执行 maxPriorityJob 作业；

Else do

调度作业 maxPriorityJob 到有空闲资源的队列中执行；

(5) 参数计算：

计算实际平均到达率 γ；

计算实际平均服务率 μ；

计算理论平均队长 L_w 和平均逗留时间 T_w；

对 L_w 和 T_w 的理论值和实际值取差值；

如果差值＞阈值，根据用户的作业特征对 γ 和 μ 这两个核心参数进行适当调节，直到 L_w 和 T_w 两个参数收敛于阈值；

(6) 利用 LATE 调度的计算方法预测作业完成时间并对进度较慢的作业启动备份任务；

} End

3.2 反馈机制设计

目前现有 Hadoop 调度算法不具有反馈机制，FIFO 以及 FAIRR & Capacity 调度都是没有反馈的调度系统。由于 Hadoop 集群中，用户的作业特征往往会发生变化，随机会出现及时性要求高的作业提交到系统中，因此非常有必要根据用户的作业特征对调度算法的相关参数进行调整，具体设计如图 2 所示。

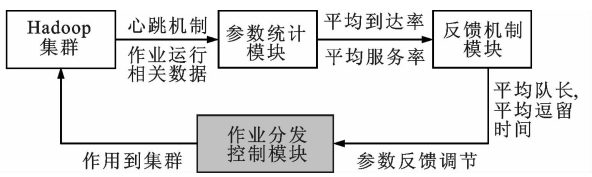

图 2 MMS 调度反馈机制设计图

图 2 中反馈机制主要包括 2 个核心模块：参数统计模块和反馈机制模块。其中，参数统计模块的输入来自于 Hadoop 集群，Hadoop 通过自身的心跳机制将相关作业运行数据，如调度时间、初始化时

间、运行时间、节点是否失效等信息传递给参数统计模块，然后参数统计模块对这些数据进行统计，获得实际的平均到达率和平均服务率这 2 个核心参数的实际值，并将其传递给反馈机制模块。反馈机制模块根据云作业调度 MMS 模型计算出平均队长和平均逗留时间的理论值，并与实际值进行对比，如果差值大于预定的阈值，则调整 MMS 调度算法的核心参数，使差值变小并收敛于阈值。

3.3　基于 LATE 的任务备份

由于 Hadoop 集群是一个异构集群环境，因此本文借鉴 LATE 调度算法来解决 Hadoop 异构环境而导致的作业运行效率问题。利用 LATE 调度算法来估计任务需要完成的时间，对那些进度低于平均进度的任务启动备份任务，并且将这些备份任务交给高效率的节点进行处理。

4　性能评估与分析

4.1　实验环境及核心参数配置

为了验证算法的有效性，本文搭建了一个测试集群作为测试平台，测试环境的硬件配置为：1 个主节点 Master，4 个从节点 Salve；CPU：Intel(R) Xeon(R) 2.0 GHz 6 核心；硬盘：300 GB SATA disks；内存：8 GB RAM。

对比实验是与 Hadoop 默认的 FIFO 调度和 FAIR 调度作对比，本文选择了 wordcount、TeraSort 和 pi 3 种不同类型的作业作为测试用例，实验核心参数配置如表 2 所示。

表 2　实验核心参数配置

实验核心参数名称	配置值
io. sort. factor	30
dfs. replication	3
dfs. block. size	134 217 728
dfs. datanode. max. xcievers	4 096
mapred. compress. map. output	True
mapred. tasktracker. map. tasks. maximum	10
mapred. tasktracker. reduce. tasks. maximum	6
mapred. map. tasks	10
mapred. reduce. tasks	6
mapred. child. java. opts	−Xmx512m

4.2　算法性能评价标准

本文采用系统的作业平均逗留时间和平均队长作为算法性能评价标准。平均逗留时间定义为平均等待时间与作业执行时间之和除以相同优先级值的作业总数目，即

$$T_w = \sum (T_q + T_e)/N_p \tag{15}$$

式中：T_q 为平均等待时间；T_e 是作业执行时间；N_p 是同一优先级作业的总数。

平均队长为

$$L_w = \sum L_t/t \tag{16}$$

式中：t 是时间，以 10 min 为一个单位，每 10 min 统计一次；L_t 是每 10 min 的平均队长。由该算法的设计思路可知，作业平均逗留时间 T_w 越小，调度算法的性能越好；同样，作业平均队长 L_w 越短，调度算法效率越高。

4.3　作业平均逗留时间验证

根据 Hadoop 系统中的作业特征，在实验时使用 3 种类型的作业进行测试。分别随机赋予同一作业不同类型的优先级权值，在一个随机的时刻提交到集群进行测试验证，根据式(15)统计作业的平均逗留时间，对比实验在相同的测试环境下切换为不同的调度器分别做实验进行测试验证，不同调度器的平均逗留时间对比如图 3 所示。

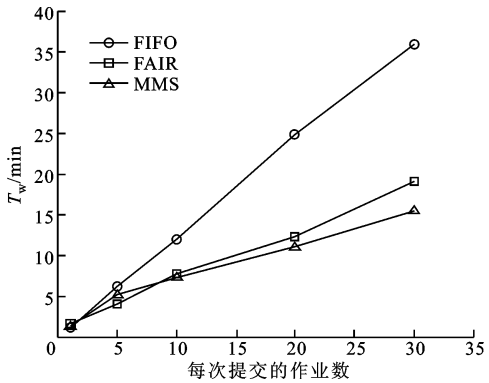

图 3　3 种调度器的平均逗留时间

图 3 中的数值是在 Hadoop 参数不变的情况下进行 10 次相同实验得到的平均值，提交作业的优先级根据随机函数计算得出。从图 3 中可以看出，当作业数目达到一定数目时，MMS 调度算法的优势就显示出来。如每次实验提交的作业总数为 30 时，MMS 的作业平均逗留时间比 FIFO 和 FAIR 分别减少了 57%和 19%，这主要是因为本文增加了反馈机制，通过反馈机制系统不断地优化 MMS 调度算法的相关参数，从而随着作业数目增加，平均逗留时间也就越短。

4.4　平均队长对比验证

本节实验环境及核心参数配置同上，根据式(16)计算平均队长，实验对比结果如图 4 所示。

图 4 中横坐标是同时运行的作业数目，对应纵

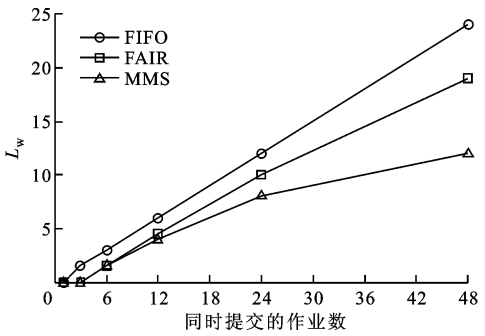

图 4 3 种调度器的平均队长

坐标的数值是在 Hadoop 参数不变的情况下进行 10 次相同实验得到的平均值。从图 4 可以看出,当作业数目增加到 48 时,MMS 的平均队长比 FIFO 和 FAIR 调度器分别减少 50% 和 37%,这是因为在 MMS 中虽然只有一个排队队列但是有多个资源池服务窗口,使得 Hadoop 集群的计算槽位资源利用率更高,而 FAIR 调度器中会出现一个队列空闲而别的队列排队很长的情况。

5 总 结

针对现有 Hadoop 经典作业调度算法在多用户、异构环境下不具备反馈机制的问题,本文基于排队论模型提出了一种云计算环境下具有反馈机制的动态作业调度算法 MMS。在本文的实验环境和实验参数下,相对于 FIFO 和 FAIR 算法,MMS 算法的平均逗留时间分别减少了 57% 和 19%,平均队长分别减少 50% 和 37%。实验结果表明,本文算法具有较好的调度效率和负载平衡能力。该算法虽然自适应性强,调度效率高,但也存在逻辑较复杂,受参数影响较大的缺点。

参考文献:

[1] 金嘉晖,罗军舟,宋爱波. 基于数据中心负载分析的自适应延迟调度算法 [J]. 通信学报, 2011, 32(7): 47-56.
JIN Jiahui, LUO Junzhou, SONG Aibo. Adaptive delay scheduling algorithm based on data center load analysis [J]. Journal on Communications, 2011, 32(7): 47-56.

[2] 左利云,曹志波. 云计算中调度问题研究综述 [J]. 计算机应用研究, 2012, 29(11): 4023-4027.
ZUO Liyun, CAO Zhibo. Review of scheduling research in cloud computing [J]. Journal of Application Research of Computers, 2012, 29(11): 4023-4027.

[3] FOSTER I, ZHAO Yong, RAICU L. Cloud computing and grid computing 360-degree compared [C]// Proceedings of the IEEE Grid Computing Environments Workshop. Piscataway, NJ, USA: IEEE, 2008: 1-10.

[4] JACKSON K R, RAMAKRISHAN L, MURIKI K. Performance analysis of high performance computing applications on the amazon web services cloud [C]// Proceedings of the IEEE International Conference on Cloud Computing Technology and Science. Piscataway, NJ, USA: IEEE, 2010: 159-168.

[5] VISHWANATH V, HERELD M, ISKRA K. Accelerating I/O forwarding in IBM blue Gene/P systems [C]// Proceedings of the IEEE International Conference on High Performance Computing, Networking, Storage and Analysis. Piscataway, NJ, USA: IEEE, 2010: 1-10.

[6] 陈康,郑纬民. 云计算:系统实例与研究现状 [J]. 软件学报, 2009, 20(5): 1337-1348.
CHEN Kang, ZHENG Weimin. Cloud computing: system instances and current research [J]. Journal of Software, 2009, 20(5): 1337-1348.

[7] JONIT N, BABA M D. First in first out (FIFO) and deficit round robin (DRR) scheduling analysis in WiMAX network [C]// Proceedings of the IEEE Control and System Graduate Research Colloquium. Piscataway, NJ, USA: IEEE, 2011: 166-174.

[8] ZAHARIA M, BORTHAKUR D, SARMA J S, et al. Job scheduling for multi-user MapReduce clusters [R]. Berkeley, CA, USA: University of California, Berkeley. Electrical Engineering and Computer Sciences Department, 2009: 1-16.

[9] TANG Zhuo, ZHOU Junqing, LI Kenli. A MapReduce task scheduling algorithm for deadline constraints [J]. Cluster Computing, 2013, 16(4): 651-662.

[10] ZAHARIA M, BORTHAKUR D, SARMA J S, et al. Delay scheduling: a simple technique for achieving locality and fairness in cluster scheduling [C]// Proceedings of the 5th European Conference on Computer Systems. New York, USA: ACM, 2010: 265-278.

[11] 陆传赉. 排队论 [M]. 3 版. 北京:北京邮电大学出版社, 2009: 64-80.

(编辑 刘杨)