第 25 卷第 3 期　　四川理工学院学报(自然科学版)　　Vol. 25　No. 3
2012 年 6 月　　Journal of Sichuan University of Science & Engineering(Natural Science Edition)　　Jun. 2012
文章编号:1673-1549(2012)03-0047-04　　DOI:10.3969/j.issn.1673-1549.2012.03.012

基于 QoS 约束的计算能力调度算法研究

丁宇光, 刘文杰, 王卫林

(辽宁大学信息学院, 沈阳 110036)

摘　要:云计算具有很强的商业性特点,以为用户提供高质量的服务为目标。针对云计算对服务质量 QoS 的需求问题以及云计算原有计算能力调度算法没有考虑用户多样性的缺点,提出了基于 QoS 约束的计算能力调度算法。该算法可以在保证为用户提供模拟的独立计算能力基础上,根据 QoS 参数生成的向量进行资源与任务的匹配,区分用户的不同服务质量需求,为用户提供符合其需求的资源。

关键词:云计算;作业调度;QoS;算法
中图分类号:TP393　　**文献标识码**:A

引　言

云计算是由并行计算、网格计算、分布式计算发展而来的[1],它将计算任务分布在大量计算机构成的机器集群经虚拟化后的资源池上,使各类用户能够根据需要获取价格低廉与性能良好的计算能力、存储空间以及各种软件服务[2],是一种通过互联网将机器集群中的各种资源经虚拟化打包成服务提供给用户的一种商业模式[3]。云计算的商业化特性要求其能满足用户多样的需求,因此,作业的调度也必须满足不同用户的不同类型作业需求。

目前,云计算下的作业调度技术研究还处于基础阶段,现有调度算法均存在一些不足。本文选取较为流行的 MapReduce 编程模型[4]进行云计算作业调度的研究。早期的 MapReduce 使用 FIFO 调度算法来进行作业的调度,其有较好的调度效率,但不能保证小作业的及时执行。Facebook 针对此缺点开发了能够保证小作业得到迅速响应,大作业保证服务水平的公平调度算法 Fair Scheduler[5]。Yahoo 开发了能够为用户提供模拟出的独立计算能力的计算能力调度算法 Capacity Scheduler[6],该算法较好的保证了用户计算能力的需求。但是因为用户的类型较少,还没有全面考虑各类用户的不同需求。

根据上述介绍算法的不足,本文引入服务质量 QoS[7]参数,根据此参数设计一种资源与任务的匹配算法,进而提出一种基于 QoS 约束的计算能力调度算法。在作业与资源进行匹配时,以 QoS 参数的匹配作为依据,选取最佳匹配的资源分配给任务。

1　云计算的主流编程模型

1.1　MapReduce 编程模型介绍

目前,大部分云计算环境都采用 MapReduce 编程模型。同时,MapReduce 不仅是编程模型还是调度模型。从 MapReduce 编程模式的计算执行过程中可抽象出三个角色: User 、Master、Worker。User 是系统的用户,负责提交作业并编写 Map 和 Reduce 函数,控制计算的方法。Master 是整个系统的核心控制器,主要功能有与用户交流、数据划分、任务调度、负载均衡、容错处理等。Worker 负责接收 Master 分配给它的任务,运行任务,并负责与 Master 的数据传输通信。一个 MapReduce 作业通常是通过 User 向 Master 提交的,Master 接到 User 的作业请求后把其加入作业队列中。Master 周期性的等待 User 通过 RPC 向其提交作业,而空闲的 Worker 周期性的通过 RPC 向 Master 发送心跳信号查询是否有任务可执行,如果有,则请求 Master 分配任务给它执行。如果 Master 的作业队列不为空,则 Worker 发送的心跳请求将会获得 Master 给它分配的任务。这是一个主动请求的任务:集群中空闲的 Worker 主动向 Master 请求任务。当 Worker 分配到任务后,通过自身调度在本地建立起任务,并执行任务[4]。图 1 是一个 MapReduce 任务请求及

收稿日期:2012-03-29
作者简介:丁宇光(1987-),男,辽宁沈阳人,硕士生,主要从事计算机网络和云计算方面的研究,(E-mail)dingyg@gmail.com

调度的过程示意图:

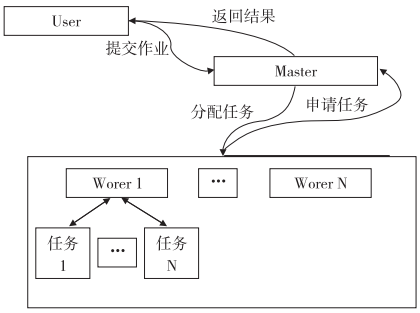

图 1 MapReduce 任务调度过程示意图

1.2 MapReduce 中的作业调度算法

目前,MapReduce 模型中主要有三种调度算法:

先进先出调度算法(FIFO):FIFO 调度算法中用户作业都被提交到唯一的一个队列内。当有空闲的 Worker 的请求任务时,Master 会按照各个作业的优先级及时间排序,再选择一个最早提交的优先级最高的作业的任务分配给该 Worker[8]。

Facebook 的公平调度算法(Fair Scheduler):公平调度算法的目标是尽可能保证各类用户都能够公平的共享集群的资源份额。当只有一个作业在系统中运行时,它将独占整个机器集群并使用所有的计算资源。而当有越来越多的作业被提交,就会有已完成任务的空闲 Worker 被释放并分配给新提交的作业,以保证各类用户都能够获得公平的共享资源[8]。

Yahoo 公司开发的计算能力调度算法(Capacity Scheduler):计算能力调度算法的核心思想是为不同用户的作业模拟出具有指定计算能力的独立的集群资源。算法分为两级调度,队列间调度时,算法会首先选择一个具有最多空闲空间的队列。在队列内选择作业的时候,采取 FIFO 算法,优先级高的作业可以优先使用队列资源。此外,算法还为了避免出现任务由于内存资源不足而造成无法执行的情况,调度算法要确保将该作业的相关任务分配到具有其所需要的内存资源的 Worker 节点上[9]。

该算法已经考虑了一部分服务质量需求,并且能够有效的处理各类型的作业。但还没有全面考虑各资源特征的分配是否满足作业多样的服务要求,因此本文提出一个基于 QoS 约束的计算能力调度算法,以期更好地满足用户多样的服务需求。

2 基于 QoS 约束的计算能力调度算法

由前文得知,现有调度算法没有全面考虑服务质量 QoS。根据用户的需求,在计算能力调度算法基础上采用 QoS 约束的匹配方法来选择适合空闲 Worker 的任务。在该方法中,QoS 参数的设计以及匹配方法是关键。

2.1 用户需求与资源节点的描述

作业需求特征:QoS 服务质量是来源于网络性能机制的参数,但在云计算中是用 QoS 表示用户所需求的各种服务特征参数。本文选取 CPU 使用率、内存使用率、网络使用率、硬盘使用率以及价格作为服务质量 QoS 参数即所需的特征变量,这组变量可以用一个一维向量表示:

$$Req = \{req_1, req_2, req_3, req_4, req_5\} \tag{1}$$

其中,$req_1, req_2, req_3, req_4, req_5$ 分别表示 CPU 使用率,内存使用率,网络使用率,硬盘使用率,价格。这些变量的值可以采用两种方法进行设置,一种是通过学习历史信息来分析作业的执行情况计算得出,另一种是由用户在提交作业时预先设定出来。本文采用较为简单的后一种方式。为了计算方便,变量的取值范围为[1,10],1 为最小值,表示对这项资源需求最少,10 为最大值,表示对这项资源需求最大。此外,由于本文算法需要避免资源的过载,引入了硬性 QoS 参数。该参数也为用户预先设定值,是用户实际所需求的内存,硬盘使用量,参数越大说明需求越高。

同时,为了体现用户对各个特征变量的偏好程度,我们使用一个权重向量,来衡量这个偏好程度:

$$k = \{k_1, k_2, k_3, k_4, k_5\} \text{ 并且 } \sum_{i=1}^{5} k_i = 1 \tag{2}$$

每个权重变量表示了用户对此 QoS 参数的重视程度。同时,还可以根据此向量进行任务的分类。

在权重向量中,哪个权重变量相对较大则将此任务划分到对应的任务分类中。节点资源特征:这组特征是与用户需求特征相对应的,也可以用一个一维向量表示:

$$Res = \{res_1, res_2, res_3, res_4, res_5\} \tag{3}$$

其中,$res_1, res_2, res_3, res_4, res_5$ 同样分别表示 CPU 使用率,内存使用率,网络使用率,硬盘使用率,价格。为了计算方便,变量的取值范围依然为[1,10],1 为最小值,表示提供这项资源的能力最小,10 为最大值,表示提供这项资源的能力最大。这些参数的取值是由节点的资源特性决定,通过心跳数据传送给 Master。同时,节点资源特征也需包含对应硬性 QoS 参数的参数,表明自身内存余量及硬盘余量,此参数越大说明节点资源使用量小,过载可能性低,反之则是越有可能过载。

2.2 作业与资源的匹配算法

在上一小节介绍的用户作业需求与资源节点的 QoS 参数向量化的基础上,可以得知作业与资源的匹配结果可以通过作业特征向量、节点资源特征向量、权重向量

计算得出。综合考虑所有 QoS 参数以及用户偏好权重的影响,资源与作业的匹配度可用公式表示如下[10-11]:

$$value = \sum_{i=1}^{5} \frac{res_i}{req_i} \times k_i \quad (4)$$

Value 值能够很好地表示出作业与资源的匹配程度,在进行特征向量匹配的同时,还需要检查硬性 QoS 参数。所谓硬性 QoS 参数就是资源必须满足作业需求的参数,不满足此要求则不能将此作业的任务分配给请求任务的 Worker。因为当不满足硬性 QoS 参数的任务分配给 Worker 后,会导致任务执行失败,或者运行时间过长,Master 监视任务执行状态后会 kill 掉此类慢任务的运行,这样就会造成不必要的浪费,同时检查硬性 QoS 参数还能避免资源节点的过载。检查方法为计算用户设置的内存需求 *mem_task*,硬盘需求 *disk_task* 与资源的内存余量 *mem_res*,硬盘余量 *disk_res*:

$$must\_value = (mem\_res - mem\_task) \times (disk\_res - disk\_task) \quad (5)$$

此外,为了使不满足硬性 QoS 参数的作业不至于饿死(始终不能获得资源),我们将规定作业优先级 priority 为0,1,2,3,4…等自然数,默认作业的优先级为0。每当作业的 $must\_value <= 0$ 时,作业的优先级调高一级即加一,这样就能保证不与有较好匹配度($value > 1$)的任务产生竞争,还能避免作业饿死的情况发生。

由此可以得出最终的匹配度 *finalvalue* 的计算公式:

$$finalvalue = \begin{cases} priority, must\_value \le 0 \\ value + priority, must\_value > 0 \end{cases} \quad (6)$$

待调度作业一般都放在队列中,同时考虑作业的优先级,某一队列内的作业与资源匹配算法流程图如图 2 所示。

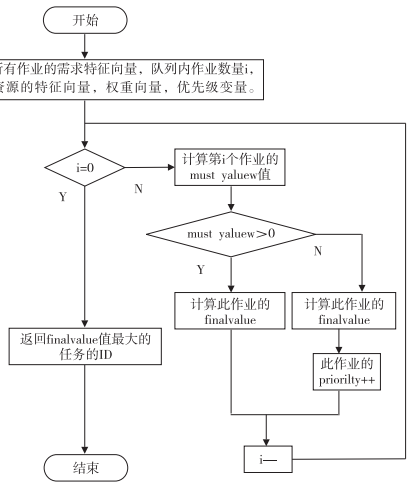

图 2　匹配算法流程图

2.3　调度算法设计

本文设计的基于 QoS 约束的计算能力调度算法核心思想就是按照用户作业需求的服务质量为用户提供集群中最合适的资源。同时考虑系统的负载问题,尽量减少选择任务进行分配时的计算量,以减轻 Master 的负载。改进原有的仅支持内存检测的方法,提出了硬性 QoS 参数,计算出 *must_value* 作为必须满足的需求标准,能够有效地预防 Worker 过载,同时,在一定程度上避免了分配任务后再终止运行不良任务带来的系统资源浪费。

根据上述思想,可以设计出调度算法的流程如下:

(1)用户提交作业到 Master,其中包括作业的配置文件,QoS 需求特征向量,权重向量,硬性 QoS 参数。Master 根据用户属性将作业放入指定队列,根据权重向量再将其放入对应的子队列,根据配置文件将相应的资源分配给队列。

(2)Worker 发送请求任务的心跳信息给 Master,其中包括资源的 QoS 特征向量,硬性 QoS 参数。Master 根据最大的 QoS 参数将资源分类,并设置一个分类参数 $i = 1$。

(3)根据队列的空闲空间从小到大将队列排序,选择最多空闲空间队列,若队列为空,通知 Worker 没有作业待分配,结束;否则执行步骤 4。

(4)根据资源的分类选择对应的子队列。若队列为空,i ++,执行步骤 5,否则执行步骤 6。

(5)根据资源特征向量的第 i 大的 QoS 参数对资源进行分类,并执行步骤 4。

(6)根据匹配算法,Master 选择子队列内一个合适的任务分配给 Worker。

(7)结束。

3　实验分析

本文提出的基于 QoS 约束的计算能力调度算法,充分考虑作业的 QoS 需求以及资源的 QoS 特征,在队列级调度中,根据最多空闲空间选择队列,在子队列级调度中,根据作业及资源的 QoS 分类选择子队列,在子队列内级调度中,设计了基于 QoS 约束的匹配算法,提高了调度的准确性。

将本文基于 QoS 的调度算法的正确匹配度与原有计算能力调度算法的比较:分别取相同数量的节点和相同数量的任务(设定 QoS 参数也相同),即图 3 中的任务数/资源数。利用 MATLAB 模拟进行任务的调度,测出本文算法的正确匹配任务数与计算能力调度算法的正确匹配任务数的对比,其结果如图 3。

从图 3 中可以看出,相比之下本文算法的匹配度有所提高。

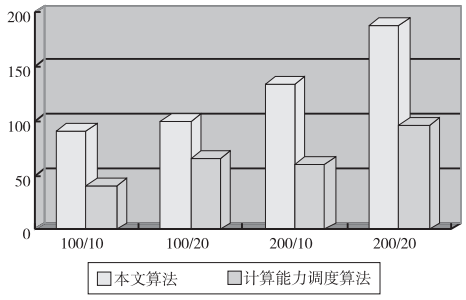

图 3 本文算法与计算能力调度算法的成功匹配任务数对比

4 结束语

本文提出一种基于 QoS 约束的计算能力调度算法。该算法在对云计算中作业及资源进行 QoS 参数向量化的基础上,把 QoS 引入到作业与资源的匹配中。采用队列、子队列、子队列内三级调度模式。既保证了准确的任务调度,还减少了调度过程的计算量。最后实验表明,该算法能较好的满足用户的多类型作业的服务需求。

参 考 文 献:

[1] 邓倩妮,陈 全.云计算及其关键技术[J].计算机应用,2009,29(9):2562-2567.

[2] Leavitt N.Is Cloud Computing Really Ready for Prime Time? [J].IEEE Computer Society Press,2009,42(1):15-20.

[3] Vquero L,Rodero-Marino L,Caceres J,et al.A break in the clouds: towards a cloud definition[J]. SIGCOMM Computer Communication Review,2009,39(1):50-55.

[4] Jeffrey D,Sanjay G.MapReduce:simplied data processing on large clusters[C]. Proceedings of OSDI'04:6th Symposium on Operating System Design and Implemention,SanFrancisco,CA.Dec.2004:1-13.

[5] Fair Scheduler for Hadoop[EB/OL],2010-04-15.http://Hadoop. apache. org/common/docs/current/Fair _ scheduler.html.

[6] Capacity Scheduler for Hadoop[EB/OL],2010-03-22. http://Hadoop. apache. org/common/docs/current/Capacity_scheduler.html.

[7] 张建勋,古志民,郑 超.云技术研究进展综述[J].计算机应用研究,2010,27(2):429-433.

[8] White Tom.Hadoop 权威指南[M].北京:清华大学出版社,2010.

[9] 王 峰.Hadoop 集群作业的调度算法[J].程序员,2009(12):119-121.

[10] 谭亚丽,于 炯,邓定兰,等.基于多维 QoS 约束的网格任务调度算法[J].计算机工程,2010,36(12):75-77.

[11] 魏正曦,陈 年,赵 攀.无线互联网的 QoS 改进技术研究[J].四川理工学院学报:自然科学版,2005,18(4):94-96.

Research on Capacity Scheduling Algorithm Based on QoS Constraints

DING Yu-guang, LIU Wen-jie, WANG Wei-lin

(College of Information, Liaoning University, Shenyang 110036, China)

Abstract: Cloud computing has a strong commercial characteristics, that provides users with high quality services. According to the demand for quality of service QoS for cloud computing and cloud computing shortcomings of existing capacity scheduling algorithm, the capacity scheduling algorithm based on QoS constraints is presentsed in the paper. Based on the guarantee to provide users with simulation independent computing capability, the algorithm can matching resources and tasks by QoS parameter generating vector, and distinguish between different user service quality requirement, provide users with its demand for resources.

Key words: cloud computing; job scheduling; QoS; algorithm