

云计算环境下虚拟机负载均衡问题研究

马华伟, 吴晓莉

(合肥工业大学 管理学院, 安徽 合肥 230009)

摘 要: 针对云计算环境下虚拟机负载均衡问题, 提出了多周期分配策略, 建立了多周期分配的虚拟机负载均衡问题的数学模型, 然后利用改进的蚁群算法进行了求解, 最后通过实验验证了算法的有效性。

关键词: 负载均衡; 资源分配; 多周期; 改进的蚁群算法

中图分类号: TP301.4

文献标识码: A

文章编号: 1000-7180(2014)04-0010-03

Research of Virtual Machine Load Balancing in the Cloud Computing Environment

MA Hua-wei, WU Xiao-li

(School of Management, Hefei University of Technology, Hefei 230009, China)

Abstract: This paper proposes a multi-cycle allocation strategy, establishes the mathematical model of the virtual machine load balancing problem in the cloud computing environment, then solves the problem with improved ant colony algorithm and finally proves the algorithm effective though an experiment.

Key words: load balancing; resource allocation; multi-cycle; Ant Colony Algorithm

1 引言

随着云计算的发展, 云计算环境下服务资源配置的负载均衡问题也越来越引人关注。很多学者对负载均衡策略进行了相关研究^[1], 但目前国内在大量的负载均衡策略的研究文献^[2-4], 主要集中于网格计算、并行计算、P2P 网络、Web 网络等领域。云计算环境下的服务资源分配是针对虚拟服务资源池中的分布式服务资源进行集中式分配, 因此云计算环境下服务资源的负载均衡是一种分布式环境下的集中式分配策略。上述的分布式环境下的策略难以直接适用于云计算环境下的服务资源分配的负载均衡问题。因此本文提出了一种多周期分配策略, 将资源池中的任务按周期进行多次分配, 避免了将所有任务一次分配而产生不必要的等待。

2 问题描述与建模

云计算环境下多周期分配的虚拟机负载均衡问题可描述如下: 云计算环境下服务资源分配的负载均衡就是在一个批处理过程中将多个任务分配给多个虚拟机, 使得在满足一定约束条件的前提下, 合理选择分配次序, 使目标函数得到优化。虚拟资源池中的多个任务, 每隔一个固定的周期被分配到虚拟机上进行服务, 每次分配的任务数等于虚拟机的数目。从资源池中有多个任务被分配到多台虚拟机, 虚拟机的总服务时间要小于最大服务时间, 一台虚拟机可分配到多个任务, 但每次只分配到一个任务, 每个任务的处理时间一定, 在满足约束条件的同时通过合理地安排任务的分配次序使目标函数得到最优, 即完成全部任务所需的多台虚拟机总服务时间的方差最小。

收稿日期: 2013-05-23; 修回日期: 2013-07-16

基金项目: 国家自然科学基金(71001032); 国防基础科研计划资助(B1420110145); 中央高校基本科研业务费专项资金资助(2012HGZY0009); 中航工业产学研专项(CXY2011HFGD20)

根据以上描述,可对云计算环境下多周期分配的虚拟机负载均衡问题模型进行以下假设:①所有的任务只能被一台虚拟机服务,且仅服务一次;②同一时间单个虚拟机只分配到一个任务;③每台虚拟机都有一定的服务时间限制,不会超过虚拟机的最大服务时间,为简化问题,假设所有虚拟机的最大服务时间相同;④每次分配的任务数等于虚拟机数目;⑤任务间的执行有先后顺序,先分配到的任务要先执行。

假设有 m 台虚拟机, n 个任务,共进行 l 次分配,每台虚拟机有最大服务时间 T , i 为虚拟机号; k 为任务号; j 表示分配次数序号; at_j 为第 j 次任务的分配时间; σ^2 为虚拟机服务时间的方差, T_i 为虚拟机 i 的总服务时间; $\bar{T}$ 为 m 台虚拟机总服务时间的平均值; t_k 为任务 k 所需的服务时间,每个任务长度除以虚拟机的处理速度; wt_{ij} 表示虚拟机 i 第 j 次分配到的任务的等待时间,若任务被分配到虚拟机时仍有任务在执行,则等待时间为在执行任务的完成时间与分配时间之差,否则为 0; bt_{iu} 为虚拟机 i 第 j 次分配到的任务的开始时间,为分配时间与等待时间之和; ft_{ij} 为虚拟机 i 第 j 次分配到的任务的完成时间,为开始时间与服务时间之和; it_{ij} 为虚拟机 i 第 j 次分配到的任务的空闲时间,若完成时间小于下一次分配时间,则空闲时间为下一次分配时间与完成时间之差,否则为 0,并引入决策变量如下:

$$x_{ij}^k = \begin{cases} 1, & \text{任务 } k \text{ 在第 } j \text{ 次分配中被分配到虚拟机 } i \\ 0, & \text{否则} \end{cases}$$

则多周期分配的 VMLBP 可描述如下:

$$\min \sigma^2 = \frac{1}{m} \sum_{i=1}^m (T_i - \bar{T})^2 \quad (1)$$

约束条件:

$$\sum_{i=1}^m \sum_{j=1}^l x_{ij}^k = 1, \forall k = 1, 2, \dots, n \quad (2)$$

$$\sum_{k=1}^n x_{ij}^k = 1, \forall i = 1, 2, \dots, m, \forall j = 1, 2, \dots, l \quad (3)$$

$$T_i = \sum_{j=1}^l wt_{ij} + \sum_{j=1}^l it_{ij} + \sum_{k=1}^n \sum_{j=1}^l t_k x_{ij}^k \leq T, \forall i = 1, 2, \dots, m \quad (4)$$

$$\sum_{k=1}^n \sum_{i=1}^m x_{ij}^k = m, \forall j = 1, 2, \dots, l \quad (5)$$

$$bt_{iu} + t_k x_{iu}^k - (1 - x_{iu}^k) T \leq bt_{i, u+1}, \forall i = 1, 2, \dots, m, \forall u = 1, 2, \dots, l \quad (6)$$

$$x_{ij}^k = 0 \text{ 或 } 1, \forall i = 1, 2, \dots, m, \forall j = 1, 2, \dots, l, \forall k = 1, 2, \dots, n \quad (7)$$

模型中式(1)为目标函数;式(2)表示一个任务只被一台虚拟机服务一次;式(3)表示单个虚拟机每次只分配到一个任务;式(4)表示最大服务时间约束,其中 $wt_{ij} = \max\{0, ft_{i, j-1} - at_{ij}\}$, $it_{ij} = \max\{0, at_j - ft_{i, j-1}\}$;式(5)表示每次分配的任务数约束;式(6)表示相邻两次任务间的时间顺序约束,其中 $bt_{iu} = at_u + wt_{iu} + it_{iu}$ 。

3 多周期分配的 VMLBP 的蚁群算法

蚁群算法(ACO)是意大利学者 Dorigo M 等研究蚂蚁寻找路径的自然行为,受到自然界中真实蚁群觅食行为的启发,进而提出的一种路径搜索算法^[5-6]。大量的研究发现,蚁群在觅食过程中通过释放信息素、感知信息素的强度来进行交流,从而选择最短搜索食物。蚁群算法是启发式算法,以其鲁棒性强、全局寻优能力强的特点在优化问题中得到广泛的应用。因此针对多周期分配的虚拟机负载均衡问题,本文提出了一种改进的蚁群算法作为求解方法。

3.1 对下一个节点的概率选择

仿效自然界的蚁群行为,蚁群算法中人工蚂蚁的行为可以描述如下:多只蚂蚁在相邻的节点间移动,从而协作异步地得到问题的解^[7]。在基本的蚁群算法中,每只蚂蚁会根据路径上的信息素,依据一定的概率选择下一个节点,距离越近的点被选择的可能性越大。本文中根据所要达到的优化目标,将概率选择公式中的距离替换为下一个任务的所需服务时间与任务分配周期的比,通过计算概率来选择下一个需要分配的任务,则可知蚂蚁对任务的选择概率应满足以下条件:(1)路径上信息素强度越强,蚂蚁到达的概率越大;(2)所需服务时间与任务分配周期越接近,蚂蚁选择到达的概率越大。

所以综上所述,设蚂蚁 k 位于虚拟机 i , $J(i)$ 表示在 i 点在满足最大服务时间约束的情况下,所能到达的需要服务的任务点集合,若 $J(i) = \emptyset$,则重新分配,若 $J(i) \neq \emptyset$,则蚂蚁 k 由任务 i 转移到任务 j 的概率为

$$p_{ij}^k = \begin{cases} \frac{\tau(i, j)^\alpha \eta(i, j)^\beta}{\sum_{j \in J(i)} \tau(i, j)^\alpha \eta(i, j)^\beta}, & j \in J(i) \\ 0, & \text{其他} \end{cases} \quad (8)$$

其中:

$$\eta(i, j) = \begin{cases} \frac{st_j}{D}, st_j > D, j \in J(i) \\ \frac{D}{st_j}, st_j < D, j \in J(i) \end{cases} \quad (9)$$

式(8)为蚂蚁 k 从任务 i 到 j 的转移概率; $\tau(i, j)$ 表示由 i 到 j 的路径上留下的信息素数量, 初始均为 1, 经过一点时间后, 当蚂蚁完成一次循环时, 由于信息素的挥发, 轨迹强度发生改变 $\tau_{ij}(t + \Delta t) = \rho\tau_{ij}(t) + \Delta\tau_{ij}(\Delta t)$, ρ 表示信息素挥发系数, $\Delta\tau_{ij}(\Delta t)$ 表示在 Δt 时间内全部蚂蚁本次循环中在路径 (i, j) 上留下的信息素的总和; $\eta(i, j)$ 为启发性信息, 这里为由 i 到 j 的路径的能见度; α 表示残留信息 $\tau(i, j)$ 的相对重要程度; β 表示预见值 $\eta(i, j)$ 的相对重要程度; 式(9)中 D 表示任务分配周期, st_j 表示任务 j 所需的服务时间。

3.2 算法描述

基于上述改进, 可得多周期分配的 VMLBP 蚁群算法流程如下:

步骤 1 初始化各参数, 设定最大循环次数 N , 计算 $\eta(i, j)$ 并存放于相应的数组中;

步骤 2 初始化任务信息, 创建空路径, 随机选取一个任务为初始点, 令当前点为 i ;

步骤 3 以当前任务 i 为出发点, 根据式(8)计算转移概率, 选取下一个任务 j , 在路径中添加任务 j ;

步骤 4 重复步骤 3, 若已遍历所有的任务, 则 $k = k + 1$, 若 $k < k + 1$, 则转步骤 2, 否则转入步骤 5;

步骤 5 每只蚂蚁得到一条以初始分配时刻为起点的路径, 每条路径就相当于所有虚拟机分配到的任务的排序, 取路径中的每 m 个任务为一次分配的任务, 分配次数 $l = 1$, 分配时间为 0, 开始时间为 0, 等待时间为 0, 计算完成时间、空闲时间;

步骤 6 $l = l + 1$, 分配时间为 $(l - 1) \cdot D$, 计算等待时间、开始时间、服务时间、完成时间, 若完成时间小于下一次任务分配时间 $l \cdot D$, 则计算空闲时间, 否则空闲时间为 0;

步骤 7 重复步骤 6, 若 $l = L$, 分配时间为 $(l - 1) \cdot D$, 计算等待时间、开始时间、服务时间、完成时间, 记录每个任务的完成时间, 若大于最大服务时间 T , 转到步骤 2;

步骤 8 计算每条路径上最后一次分配的 m 个任务完成时间的方差, 并保存方差最小的路径, 在该路径上进行信息素更新;

步骤 9 判断循环次数 n 是否达到 N , 若 $n >$

N , 则终止运算, 输出上一步迭代的最优结果, 否则转到步骤 2。

4 实验结果与分析

本文提出的改进的蚁群算法采用 C++ 语言在 Windows XP(P4 2.8 GHz, 1 GB) 平台上编程实现, 为了验证算法的性能和有效性, 随机选取一个任务为初始点, 构造一个包含 9 个任务的分配网络(设分配周期 $D = 80$ 单位时间, 虚拟机数量 $m = 3$, 分配次数 $l = 3$, 虚拟机最大服务时间 $T = 300$ 单位时间), 然后利用此数据对算法进行测试。

在蚁群算法中设定残留信息的相对重要程度 $\alpha = 1$ 、预见值的相对重要程度 $\beta = 2$ 和信息素的挥发系数 $\rho = 0.5$, 运用 10 只蚂蚁, 迭代 100 次获取结果的最优值。为了进一步分析算法执行的稳定性, 本文选择遗传算法进行比较, 相关参数设置如下: 种群大小 $\text{popsize} = 5$, 交叉率 $p_c = 0.1$, 变异率 $p_m = 0.01$, 最大世代数 $\text{generation} = 100$, 并将得出的结果与蚁群算法得到的结果进行比较, 可见表 1。

表 1 蚁群算法与遗传算法结果比较

运行次数	ACO		GA	
	方差	CPU	方差	CPU
1	0.227 2	0.031	0.227 2	0.046
2	0.312 2	0.031	0.312 2	0.062
3	0.512 7	0.031	0.526 3	0.062
4	0.526 3	0.031	1.098 4	0.062
5	0.925 0	0.046	1.100 7	0.046
6	1.098 4	0.031	1.601 1	0.078
7	1.100 7	0.046	1.671 5	0.062
8	1.654 2	0.031	1.767 9	0.078
9	1.671 4	0.046	1.909 1	0.062
10	1.975 3	0.046	1.974 2	0.078
平均值	1.000 3	0.037	1.218 8	0.063

由表 1 可见, 在 10 组数据上蚁群算法得到的结果的平均值更小, 且执行时间的平均值更小。比较表 1 的第 2 列和第 4 列可以看出, 在目标函数值相同时, 蚁群算法的执行时间更短。由此可见, 相对于遗传算法, 蚁群算法的平均性能更优, 即考虑了时间因素的改进蚁群算法具有更好的全局寻优能力, 更易于找到较优的解。由表 1 的第 3 列可以看出, 10 组实验数据算法执行时间均在 0.05 s 以内, 总的平均执行时间为 0.037 s。由此可见, 本文提出的改进蚁群算法可以用于求解云计算环境下的虚拟机负载均衡问题, 且可在较短的时间内取得满意的结果。

(下转第 17 页)

5 结束语

本文以省级档案馆的数字化建设为背景,设计了面向多租户可定制的管理系统模型,并从多租户存储模型和业务逻辑的可定制框架两个方面对该模型进行分析和阐述.该模型以较为低廉的成本,支持了各级档案馆的租户,同时可以验证业务逻辑的正确性,通过组件化的服务防止异常情况的发生.

进一步的研究工作主要集中在如何扩展服务规则管理模块,包括如何定义提供数据服务的方法和制定标准 API 接口,以便支持一定范围内的租户自定义处理逻辑.

参考文献:

- [1] Zhang Xiangyu, Ai Jing, Wang Zhongyuan, et al. An efficient multi-dimensional index for cloud data management [C]//Proceedings of the CloudDB. HongKong, China, 2009:17-24.
- [2] Shi Yuliang, Luan Shuai, Li Qingzhong, et al. A multi-tenant oriented business process customization system [C]//New Trends in Information and Service Science.

(上接第 12 页)

残留信息的相对重要程度 α 、预见值的相对重要程度 β 和信息素的挥发系数 ρ ,体现了相关信息痕迹、预见度及挥发对蚂蚁决策的相对影响.为了进一步分析各参数对解的影响,我们选用不同的 α 、 β 、 ρ 取值.实验表明, α 、 β 、 ρ 的取值变化均会影响问题的求解结果,多次运行后得出本文的最佳参数设定为 $\alpha=1$, $\beta=2$, $\rho=0.5$,此时得到的实验结果最优,且收敛速度最快,具有较强的全局寻优能力.

实验结果表明本文针对云计算环境下的虚拟机负载均衡问题建立的模型是可解的,算法是有效的.由表 1 可以看到由改进的蚁群算法得到的结果更优,而且具有更快的收敛速度,算法的性能更优.

5 结束语

本文对云计算环境下的虚拟机负载均衡问题进行了研究,建立了数学模型,并针对其中特定约束条件进行了研究,最后通过蚁群算法进行了求解,对指导现实生活中的实际情况具有一定的理论意义.

参考文献:

- [1] Li Yajun, Yang Yuhang, Ma Maode, et al. A hybrid

Gyeonqiu, 2009.

- [3] 刘莹. 基于 XML 的数字图书馆应用模型设计[D]. 济南:山东大学, 2011.
- [4] 管永明. 多租户软件中动态个性化配置与定制技术研究[D]. 济南:山东大学, 2011.
- [5] 张坤,李庆忠,史玉良. 面向 SaaS 应用的数据组合隐私保护机制研究[J]. 计算机学报, 2010(11): 2044-2054.
- [6] 李晓娜,李庆忠,孔兰菊,等. 基于共享模式的 SaaS 多租户数据划分机制研究[J]. 通信学报, 2012, (S1): 110-120.
- [7] 孔兰菊,李庆忠,史玉良. 面向 SaaS 应用基于键值对模式的多租户索引研究[J]. 计算机学报, 2011(11): 2039-2047.

作者简介:

吴振宇 男, (1988—), 硕士研究生. 研究方向为电子服务、云计算.

吴健 男, (1962—), 教授. 研究方向为软件工程、服务计算、电子服务、分布式计算、语义 WEB.

张涛 男, (1976—), 副教授. 研究方向为软件安全性、航空网络软件技术、软件测试.

load balancing strategy of sequential tasks for computational grids[J]. Future Generation Computer Systems, 2009, 25(8): 819-828.

- [2] 王少峰,周忠,吴威. 一种面向分布式虚拟环境的分层迭代负载均衡算法[J]. 软件学报, 2008, 19(9): 2471-2482.
- [3] 汪永琳. 超节点结构 P2P 中负载均衡的信息索引机制[J]. 计算机工程与科学, 2009, 31(8): 107-109.
- [4] Amaro de Sousa, Dorabella Santos, Pedro Matos, et al. Load balancing optimization of capacitated networks with path protection[J]. Electronic Notes in Discrete Mathematics, 2010, 36(1): 1249-1256.
- [5] Dorigo M. Optimization, learning and natural algorithms [D]. Italy: Politecnico di Milano, Department of Electronics, 1992.
- [6] Dorigo M, Caro G Di, Gambardella L M. Ant algorithms for discrete optimization[J]. An artificial Life, 1999, 5(2): 137-172.
- [7] 邢文训,谢金星. 现代优化计算方法[M]. 2 版. 北京:清华大学出版社, 2005.

作者简介:

马华伟 男, (1977—), 博士, 副教授. 研究方向为车辆调度问题、运筹与优化.