第 38 卷 第 2 期
Vol.38 No.2

计 算 机 工 程
Computer Engineering

2012 年 1 月
January 2012

·人工智能及识别技术· 文章编号：1000—3428(2012)02—0166—03 文献标识码：A 中图分类号：TP301.6

异构分布式系统的负载均衡调度算法

杨 锦，李肯立，吴 帆

(湖南大学计算机与通信学院，长沙 410082)

摘 要：提出一种异构分布式系统的负载均衡调度算法。对异构系统进行建模，使用染色体建立任务集合调度模型，根据该模型制定适应度函数，将其作为衡量负载均衡的标准，利用该标准对异构系统进行任务调度，并动态设定最大进化代数，以此改进动态遗传算法。实验结果表明，该算法具有较好的负载均衡性能。

关键词：异构系统；变异操作；负载均衡；遗传算法

Load Balance Schedule Algorithm for Heterogeneous Distributed System

YANG Jin, LI Ken-li, WU Fan

(School of Computer and Communication, Hunan University, Changsha 410082, China)

【Abstract】The classic Genetic Algorithm(GA) limits the evolution because the next generation cannot inherit the most adaptable chromosome. To improve the algorithm, this paper proposes a dynamic genetic algorithm. It creates a model for the heterogeneous system, and formulates criterion for measuring load balance according to the model, then uses the formulated criterion in scheduling jobs on the heterogeneous system. The algorithm allows configuring the maximum evolution generation dynamically. Experimental results show that the improved algorithm has better load balance performance.

【Key words】heterogeneous system; mutation operation; load balance; Genetic Algorithm(GA)

DOI: 10.3969/j.issn.1000-3428.2012.02.054

1 概述

随着计算机技术的发展，异构分布式系统日益受到人们的重视因而被普遍应用。但是由于异构系统中各个节点的处理能力和容量不均衡，作业的到达模式不一致，从而造成有的节点大量的资源被闲置、有的节点却负载过重，这就要解决异构系统下负载均衡的问题。为使异构系统要获得良好的负载均衡，就需要设计良好的负载均衡任务调度器，那么问题的关键就取决于负载均衡调度算法。

异构系统中基于负载均衡的调度问题，现在应用最多的算法都是基于 RR(Round Robin)算法和遗传算法(Genetic Algorithm, GA)，具体如下：

(1)Round Robin 算法

Round Robin 算法是一种简单的快速调度方法，将所提交的任务集合按照顺序分配给所有的服务节点。Round Robin 算法的优点是调度速度快，但是缺点就是没有考虑服务节点的负载情况。

(2)遗传算法

遗传算法是将任务序列和机器序列的一个映射看成是一个染色体。首先产生一个初始种群，通过选择适应度高的染色体进行交叉，变异等操作之后产生新的种群，新的种群具有更好的适应度。文献[1-3]提出用遗传算法来解决异构分布式系统中的调度问题。遗传算法随机生成初始种群，为提高效率，文献[4-5]提出启发式算法 LJFR(Longest Job to Fastest Resource)来产生初始种群。文献[6]提出 MGA(Messy Genetic Algorithm)。

遗传算法虽然在任务调度算法中应用很广泛，但是，由于现有的遗传算法都要求设置最大进化代数，因此会限制种群的继续进化。而且在遗传算法的进化过程中，交叉操作和变异操作可能会使得这一代种群的适应度最高的染色体遗传不到下一代。针对上述缺陷，本文提出一种改进的动态遗传算法。

2 模型的建立和问题的形式化

2.1 异构系统调度模型

以一个由 n 个性能不同的处理器所构成的异构分布式系统处理由 m 个不同用户所提交的 m 类独立非抢占式任务[7]。m 和 n 都是大于或等于 1 的正整数。令 $N=\{p_1,p_2,\cdots,p_n\}$ 表示异构系统服务节点的集合；令 $C=\{c_1,c_2,\cdots,c_n\}$ 表示 N 所对应每个节点的计算能力，其中，p_i 代表 c_i 的计算能力。则该异构分布式系统的总计算能力 TC 可以表示为：

$$TC=\sum_{i=1}^{n}c_i \tag{1}$$

m 类用户提交 m 种类型的任务，令 $T=\{t_1,t_2,\cdots,t_m\}$ 表示 m 类任务的集合，其中，t_i 为第 i 类用户提交的任务。假设任务 t_i 的到达服从 $\lambda=\lambda_i$ 的泊松分布，则有单位时间内任务到达数目为 λ_i；假设任务 t_i 的计算量服从 $\lambda=w_i$ 的泊松分布，则有任务 t_i 的平均计算量为 w_i。由上述介绍可以得出，在单

基金项目：国家自然科学重大研究计划基金资助项目(90715029, 60603053)

作者简介：杨 锦(1985－)，男，硕士，主研方向：分布式系统，任务调度；李肯立，教授、博士生导师；吴 帆，博士

收稿日期：2011-07-20 **E-mail：**hnuyj@163.com

位时间内，任务 t_i 的平均计算量 TW_i 为：

$$TW_i = w_i \times \lambda_i \tag{2}$$

单位时间内到达的所有类型任务的平均计算量 TW 为：

$$TW = \sum_{i=1}^{m}(w_i \times \lambda_i) \tag{3}$$

2.2 染色体设计

用染色体给任务集合的一个调度建立模型，图 1 为染色体模型。图 2 为 9 个任务调度与 3 个服务节点的映射关系。图 3 为交叉操作后的染色体，交叉操作是随机选择一个点作为 2 个染色体的交叉点。如图 3 选择交叉点为 4。图 4 为交叉操作后新的后代染色体。其中，图 2 和图 3 的竖粗线代表染色体交叉位置。

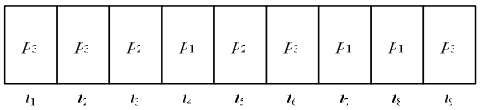

图 1 染色体模型

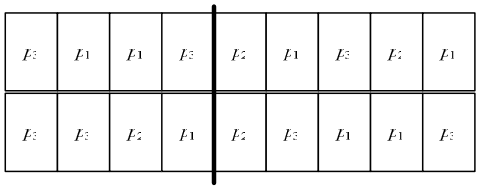

图 2 9 个任务调度与 3 个服务节点的映射关系

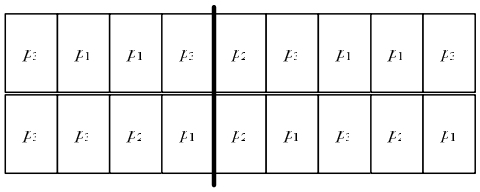

图 3 交叉操作后的染色体

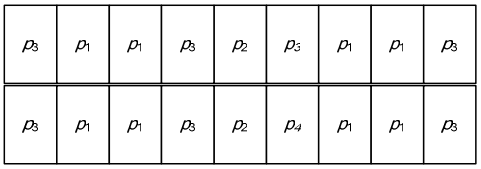

图 4 交叉操作后新的后代染色体

由图 4 可知，变异操作是按照一定的概率对染色体中的某个单元替换为其他同类型的单元。

2.3 优化目标

在异构系统中，需要根据节点的计算能力不同而进行任务调度。将节点 p_i 的负载表示为：

$$LB_i = \frac{\sum_{j=1}^{m}(p_{ij} \times TW_j)}{c_i} \tag{4}$$

其中，p_{ij} 表示第 j 类任务被分配到第 i 个计算节点上的概率，当第 j 类任务被分配到第 i 个计算节点上时，$p_{ij}=1$，否则 $p_{ij}=0$；在理想情况下，当系统的负载完全平衡时，有下式成立：

$$LB_i \equiv LB_j \tag{5}$$

其中，$1 \leqslant i$； $j \leqslant n$。也就是说当所有节点的负载相等时，达到负载均衡的理想状况。而对于整个异构系统来说，负载总是不变的：

$$LB = \frac{TW}{TC} = \frac{\sum_{j=1}^{m}(w_j \times \lambda_j)}{\sum_{k=1}^{n} c_k} \tag{6}$$

在理想情况下，达到负载均衡的条件是 $\forall i \in Z \wedge i \in [1,n]$，都有：

$$LB_i = LB \tag{7}$$

然而在实际的任务调度过程中，由于单个任务的独立性和不可分割性，因此使得要达到理想的负载均衡几乎是不可能实现的。该文在假设任务已经分配好的情况下，采用 LB 的标准差来衡量系统负载均衡性能。用 τ_i 来表示节点 i 上的负载均衡标准差，表示为：

$$\tau_i = \left| LB - LB_i \right| \tag{8}$$

则异构系统的负载均衡标准差形式化表示为：

$$\tau = \sqrt{\sum_{i=1}^{n} \tau_i^2} \tag{9}$$

τ 值越小，异构系统的负载越均衡。当 τ 值为 0 时，系统就达到理想的负载均衡状态。目标是尽量使得 τ 值最小化。

3 算法描述

3.1 改进的动态遗传算法

本文通过增加对 τ 值的动态阈值设定的方法改进遗传算法，提出一种以负载平衡为目的的改进的动态遗传算(Improved Dynamic Genetic Algorithm, IDGA)。IDGA 算法步骤如下：

(1)生成初始种群。对每个染色体计算 τ 值和适应度函数 F。记所有染色体中 τ 值最小的为 $\tau_{\min}$，设置阈值 $\tau_c = \tau_{\min}$，用 $s_{\min}$ 记录下 $\tau = \tau_{\min}$ 时的调度。

(2)选择操作。根据适应度函数 F 选择染色体直接遗传到下一代或者通过交叉操作产生新的染色体遗传到下一代。

(3)交叉操作。在进行交叉操作时，遗传算法的核心所在为 2 个染色体通过交叉操作产生 2 个新的染色体遗传到新的种群。

(4)变异操作。变异操作是模仿基因在遗传过程中存在变异的情况，按照一定的概率使染色体产生突变。从而产生新的个体进入下一个种群。

(5)判断操作。对每个染色体计算其 τ 值和适应度函数 F。记所有染色体中 τ 值最小的为 $\tau_{\min}'$。如果 $\tau_{\min}'$ 值小于阈值 τ_c，用 $s_{\min}$ 记录 $\tau = \tau_{\min}'$ 时的调度，并且将已经产生种群的代数设置为零，给定一实数 $\mu \in (0,1)$，如果 $\mu \times \tau_c > \tau_{\min}$，则设置 $\tau_c = \tau_{\min}$，否则 $\tau_c = \mu \times \tau_c$，然后跳到步骤(2)继续执行算法；如果 $\tau_{\min}'$ 值大于或等于阈值 τ_c 并且种群代数还没有达到最大，则跳到步骤(2)继续执行算法；如果 $\tau_{\min}'$ 值大于或等于阈值 τ_c 并且种群代数已经达到最大，则 $s_{\min}$ 为当前情况下的最优调度，算法结束。

3.2 适应度函数

适应度函数反映的是染色体对于环境的适应程度，其值越高，染色体被选择的可能性就应该越大。显然，在本文的模型中，当所有的任务全都分配给异构系统中计算能力最差的节点时，系统的负载均衡达到最差。那么，可以确定 τ 的取值范围。令：

$$\gamma = \sqrt{\frac{1}{n}\left((n-1)\left(\frac{\sum_{j=1}^{m}(w_j \times \lambda_j)}{\sum_{k=1}^{n} c_k}\right)^2 + \left|\frac{\sum_{j=1}^{m}(w_j \times \lambda_j)}{\sum_{k=1}^{n} c_k} - \frac{\sum_{j=1}^{m}(w_j \times \lambda_j)}{c_{\min}}\right|^2\right)} \tag{10}$$

可以确定 τ 的取值范围为：

$$0 \leqslant \tau \leqslant \gamma \tag{11}$$

其中，$c_{\min}$ 为异构系统中计算能力最差的节点的计算能力。对于一个染色体所对应的调度 s 的适应度函数：

$$F(s)=\gamma-\tau_s \tag{12}$$

其中，τ_s 为调度 s 所对应的 τ 值。τ 值越小，适应度函数就越大，表示负载均衡性能就越好。

4 实验结果与分析

引入一个变量 RT 来表示系统的响应时间，RT_i 为节点 i 的响应时间，分析式(4)，发现 $\sum_{j=1}^{m}(p_{ij}\times TW_j)$ 代表分配到节点 i 上的任务量，而 c_i 代表节点 i 的计算能力。令 $\varsigma \in (0,+\infty)$，则：

$$RT_i=\varsigma\frac{\sum_{j=1}^{m}(p_{ij}\times TW_j)}{c_i} \tag{13}$$

再引入一个变量 ε 表示异构系统的利用率，则：

$$\varepsilon=\frac{\sum_{j=1}^{n}RT_i}{n\times \mathrm{Max}_{1\leqslant i\leqslant n}\{RT_i\}} \tag{14}$$

其中，ε 的数学意义为节点的有效利用时间与总时间的比例。实验参数设置如表 1 所示。

表 1 实验参数设置

参数	取值范围
μ	0.9, 0.8, 0.7, 0.6, 0.5
m	100, 200, 500, 1 000
n	20
λ_i	0.2, 0.4, 0.5, 0.8, 1.0
w_i	(5, 20)
c_i	(10, 20)
G	50

图 5 为采用本文算法，当 m 的取值分别为 100、200、500 和 1 000 时系统利用率对比(限于篇幅，罗列部分数据)。

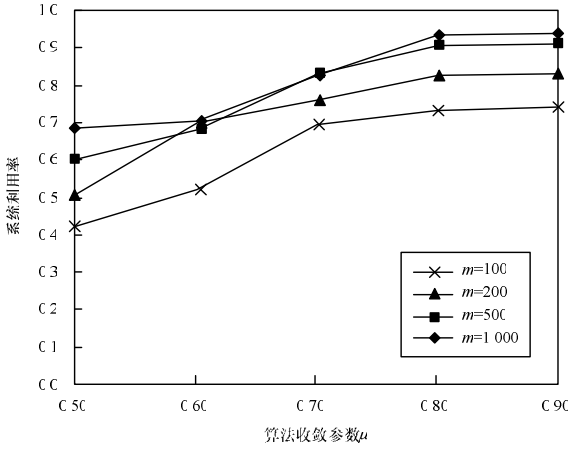

图 5 不同情况下本文算法系统利用率对比

由图 5 可知，在尽量提高算法效率的前提下 μ 取值为 0.8 时能达到较好的效果。

图 6 为当 m=1 000 时，3 种算法的系统利用率对比。由图 6 可知，本文算法比 RR 算法、GA 算法具有更好的负载均衡性能。

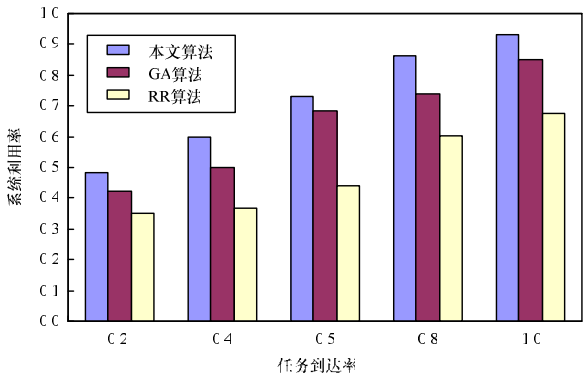

图 6 m=1 000 时 3 种算法系统利用率对比

5 结束语

本文对异构分布式环境下的调度问题进行建模，通过将系统中负载标准差作为衡量异构系统负载均衡的标准，提出一种改进的动态遗传算法。实验结果表明，该算法和 RR 算法和 GA 算法进行比较，发现其能够达到更好的负载均衡。由于遗传算法的时间复杂度比较高，因此今后将对本文算法建立并行模型，以提高效率。

参考文献

[1] Zomaya A Y, Teh A Y. Observations on Using Genetic Algorithms for Dynamic Load-balancing[J]. IEEE Transactions on Parallel and Distributed Systems, 2001, 12(9): 899-911.

[2] Martino D, Mililotti M. Sub Optimal Scheduling in a Grid Using Genetic Algorithms[J]. Parallel Computing, 2004, 30(5/6): 553-565.

[3] Ndrew P, Thomas N. Framework for Task Scheduling in Heterogeneous Distributed Computing Using Genetic Algorithms[C]// Proc. of the 15th Artificial Intelligence and Cognitive Science Conference. Mayo, Ireland: [s. n.], 2005.

[4] Abraham, R. Buyya, B. Nath. Nature's Heuristics for Scheduling Jobs on Computational Grids[C]//Proc. of the 8th IEEE International Conference on Advanced Computing and Communications. [S. l.]: IEEE Press, 2000.

[5] Avier C, Fatos X, Abraham A. Genetic Algorithm Based Schedulers for Grid Computing Systems[J]. International Journal of Innovative Computing, Information and Control, 2007, 3(5): 1053-1071.

[6] Mohammadzadeh J, Moeinzadeh M H, Sarah S R, et al. Scheduling Dynamic Load-balancing in Parallel and Distributed Computers Using Modified Genetic Algorithm with Time Dependent Fitness Function[C]//Proc. of IEEE International Conference on Intelligent Computing and Intelligent Systems. Shanghai, China: IEEE Press, 2009.

[7] Qin Xiao, Xie Tao. An Availability-aware Task Scheduling Strategy for Heterogeneous Systems[J]. IEEE Transactions on Computers, 2008, 57(2): 188-199.

编辑 刘 冰