

基于改进遗传算法的异构环境混合调度策略研究

范 敏, 胡 伟

(绵阳师范学院, 四川 绵阳 621000)

摘 要: 异构环境下任务调度是 NP 问题, 它关注大规模的资源 and 任务调度, 要求采用的调度算法能够具有高效性. 随着任务数和资源数的增加, 遗传算法表现出慢速收敛的缺点. 为了克服其缺点, 在改进的遗传算法的基础上, 增加了分组和负载均衡处理策略, 提出了一种混合遗传调度策略. 仿真实验表明, 基于改进遗传算法的混合调度策略比传统的调度策略性能更优, 其算法更符合复杂的异构环境, 能更好满足系统的时间特性和最小化资源开销的问题.

关键词: 任务调度; 遗传算法; 异构性; 调度策略

中图分类号: TP391

文献标识码: A

文章编号: 1000-7180(2010)08-0119-05

Research of Hybrid Scheduling Strategy in Heterogeneous Environment Based on Improved Genetic Algorithm

FAN Min, HU Wei

(Mianyang Normal University, Mianyang 621000, China)

Abstract: Task scheduling in heterogeneous environment is a NP problem, it is concerned about the large-scale resource and task scheduling, requires the scheduling algorithm is highly efficient. As the number of tasks and resources to increase the number of genetic algorithm to show the shortcomings of slow convergence. In order to overcome its shortcomings, this paper improved genetic algorithm based on the increased packet processing and load balancing strategy, a hybrid genetic scheduling policy, simulation experiments show that the hybrid genetic algorithm-based scheduling policy than the traditional scheduling strategy better performance, the algorithm more complex, heterogeneous environment, the system can better meet the characteristics of time and resources to minimize the overhead issue.

Key words: task scheduling; genetic algorithm; heterogeneity; scheduling strategy

1 引言

异构环境下的任务调度是 NP 问题, 而找到更好的调度策略, 一直是研究的热点问题^[1]. 遗传算法在求解 NP 问题时, 能够得到很好的解, 目前关于遗传算法用在异构环境下任务调度的研究成果很多, 如传统的遗传算法. 和模拟退火、蚁群等算法相比, 遗传算法能够得到更优的解, 但遗传算法有其天生的缺点: 收敛速度缓慢, 早熟收敛, 缺乏爬山能力, 其中早熟收敛最为严重. 本文提出了一种基于改进遗传算法的混合调度策略, 此调度策略首先将任务

进行分组, 然后对每一组任务用遗传算法来进行调度处理, 在调度过程中, 使用负载均衡方法来调整染色体中任务的分布. 分组过程减少了染色体编码的长度, 从而加快了遗传算法的收敛速度; 负载均衡处理使染色体更加优良, 从而也加快了遗传算法的收敛速度^[2].

2 相关工作

在目前提出的异构环境下调度算法中, 没有一个被证明是最优的, 所以这些调度算法都是近似的算法. 对一些调度算法来说, 其收敛速度快, 像网络

中的 Min-Min、QoS Min-Min、Max-Min、基于 QoS 的混合 Min-Min 等,一般能获得问题的近似最优解,但多数是考虑单任务调度算法,要么就是采用的调度算法不具有高效性.下面简单介绍几种已经出现的经典调度策略^[3].

Min-Min 算法实现简单,执行时间快.算法的思想是:首先映射小的任务,并且映射到执行快的机器上.执行过程为:计算要参与映射事件的每个任务在各个机器上的期望完成时间,找到每个任务的最早完成时间及其对应的机器;从中找出具有最小完成时间的任务,将该任务指派给获得它的机器;指派完成后,更新机器期望就绪时间并将已完成映射的任务从任务集合中删除,重复上面的过程,直到所有的任务都被映射完,该算法的作用是让具有最小完成时间的任务优先调度,且简单快速,但导致负载的不平衡.

QoS Min-Min 异构分布式系统任务调度策略,此调度算法是在 Min-Min 调度算法的基础上,增加了通信延迟的计算,任务是否能够在系统中某台计算机上执行的判断,任务在所分配计算机中位置的计算,和 Min-Min 调度算法相比,QoS Min-Min 算法减少了任务的淘汰率和提高了负载平衡能力^[4].基于 QoS 的混合 Min-Min 任务调度策略,根据任务性质和系统自身状态,调度策略采用了在线模式和批模式的映射方法,从而使系统 CPU 能够得到充分利用,并且能够使系统处理更多的任务.和 Min-Min 调度算法相比,极大地减少了任务淘汰率和提高了系统 CPU 的利用率;和 QoS Min-Min 调度策略相比,可以更好满足紧急任务和 QoS 时间短的任务的处理,但是此调度策略适应环境的能力不够强.

3 基于改进遗传算法的异构环境混合调度策略

为了找到更好的调度算法,本文将遗传算法进行改进,然后再将改进的遗传算法和其他方法进行结合.遗传算法应用在任务调度问题时,如果需要调度的任务很多,则染色体的长度就会很长,并且使得解空间变得很大,此时遗传算法的收敛速度会很慢;如果将问题分成多组来求解时,染色体长度会变小,对每一组来说,遗传算法收敛速度也会明显下降.但如果分组过多,每一组中包含的任务数量少,一方面,对每一组来说,解是次优解,但对整个调度问题来说,则会影响整体解的优劣性;另一方面,分组过多,每一组都需要经历遗传算法的各个阶段,而整个

问题求解时间是所有分组求解时间的叠加,从而影响算法收敛时间^[5].本文给出了合理的分组数,使得解在达到次优的情况下,收敛速度最快.染色体的选择对于遗传算法收敛速度也会产生影响,优良的染色体会加快算法收敛速度.为了使染色体有利于遗传算法向收敛方进化,本文采用负载平衡方法使染色体朝优良方向进化,其思想是:在变异处理后,利用负载平衡方法,调整染色体中任务的分布,使得染色体中各台计算机的负载更加平衡.

3.1 任务分组

当使用遗传算法将大量任务分配给系统时,首先要进行编码,为了使染色体能够包含所有解空间,则染色体的长度很长,从而导致算法收敛速度减慢,但如果将求解的问题分成多组,这样就可以减少染色体的编码长度,对一组任务来说,提高了遗传算法的收敛速度^[6].为了提高整个问题的分解速度,这里分组的策略是:

(1) 对所有任务按粒度进行排序,并且将这些任务按大小次序进行编号,进而为遗传算法服务.

(2) 将编号后的任务发成几组.任务分成的组数和任务数量、计算机数量、每台计算机平均分到的任务数量有关,用以下分式来进行求解.

$$G = T / (C * A) \quad (1)$$

式中, T 表示所有任务, C 表示参与计算的计算机, A 表示每台计算机分到的任务数量,其中 T 、 C 由系统决定, A 由用户自己给定,一般情况下为 8 ~ 128 较为合理.如果太小,对于给定的任务来说,分组数量太多,如果大于 128 则分组的数量减少.

(3) 将任务粒度最大的一组首先提交给遗传算法进行调度,然后逐步将粒度小的任务组提交给遗传算法进行调度.

3.2 改进的遗传算法

遗传算法是一种基于空间搜索的算法,它通过选择、交叉、变异以及适者生存的理论,模拟自然进化过程来寻找所求问题的答案^[6].因此,遗传算法的求解过程也称优化过程.根据异构环境下任务调度的特点,下面对遗传算法进行改进.

(1) 编码方法

对于异构环境下的任务调度来说,是将不同任务分配给各台计算机,为了将任务和计算机之间的映射关系反映到遗传算法解的形式中,本文采用字符串编码方案.其特征为:用 0 来隔开各台计算机之间的任务分配方案,各台计算机中的任务用任务号来标记.

(2) 染色体和群体的初始化

采用随机方法生成染色体,每台计算机中的任务个数是随机生成的,但各台计算机的任务数量之和为所需要调度的任务数量,任务也是随机分布在不同计算机中^[7].

(3) 适应度函数

采用相对负载平衡处理方法来计算染色体的适应值.用 $R_1, R_2, \dots, R_m$ 表示在分配任务前各台计算机的负载情况,其中 m 表示计算机的台数,当继续往不同计算机上分配任务时,考虑了当前各台计算机的负载,这样使得各台计算机能够尽量保持负载平衡,并且使得系统效率更高.当分配 N_i 个任务到达计算机 i 时,计算机 i 需要的计算时间用 S_i 表示, S_i 的值可以用式(2)来计算.

其中, t_{ik_j} 表示任务 k_j 在计算机 i 上的执行时间, $\text{Com}_{k_j i}$ 表示任务 k_j 到计算机 i 的通信时间,如果任务执行时间和通信时间可以重叠,则通信时间可以看成 0.

$$S_i = R_i + \sum_{k_j=0}^{N_i} t_{ik_j} + \sum_{k_j=0}^{N_i} \text{com}_{k_j i} \quad (2)$$

$$L_j = (t - ST_j) \cdot C_j \quad (3)$$

$$R = \sqrt{\frac{\sum_j (Tt_j - t)^2}{N}}, j \in s, N \quad (4)$$

并且 N_i 必须满足:

$$N = \sum_{i=1}^m N_i \quad (5)$$

式中, m 表示计算机的台数, i 为 $1 \sim m$. 此时两台计算机负载差的绝对值可用式(6)来计算.

$$P_i = |S_{(i+1) \% m} - S_i| \quad (6)$$

此时利用 E_i 表示染色体 i 的适应值,其值可以利用式(7)来计算.

$$E_i = \sum_{m=1}^m P_m \quad (7)$$

根据负载平衡的思想, E_i 的值越小,各台计算机之间的负载越平衡,则染色体越优良.

(4) 选择方法

采用轮盘赌的方法,从式(7)可以知道, E_i 越小,染色体越优良.为了使优良染色体被选择概率越大,这里需要对 E_i 执行相应变换,首先用式(8)进行变换.

$$F_i = \frac{1}{E_i} \quad (8)$$

E_i 的值越小,则 F_i 的值越大,染色体越优良.为

了使用轮盘赌方法,利用式(9)进一步变换,根据式(9)可以知道 δ_i 之和为 1, δ_i 值越大,在轮盘赌中占的百分比越大,从而被选择的概率越大.

$$\delta_i = \frac{F_i}{\sum_{j=1}^m F_j} \quad (9)$$

(5) 交叉方法

使用循环交叉方法,设有号码为 $1 \sim 9$ 的任务需要被分配到 4 台计算机中,在遗传算法执行过程中,假设有两个编码 A 和 B ,其编码如下:

$$A = 368015079024$$

$$B = 920578064031$$

为了使循环处理更加方便,首先删除各个编码中的 0,并且记住它们的位置.可以得到两个新的编码.

$$A' = 368157924$$

$$B' = 92786431$$

然后在两个编码中取一个固定位置的值,这里是编码中的第一位.

$$A'' = 3 - - - - -$$

$$B'' = 9 - - - - -$$

在 A'' 中寻找 B'' 中的 9,然后在 B'' 中找到与 A'' 中 9 对应位置的值 4,继续在 A'' 中找到 4,重复上面的过程,直到在 B'' 中找到值为 3 的任务,这时,如果再寻找下去会出现循环象,所以终止寻找.

当循环终止后,将两个编码中没有查找出来的编码进行交换,得到新的编码,这种编码方法就是循环编码.新的编码结果如下:

$$A' = 365187924$$

$$B' = 928756431$$

最后根据编码的特点,将计算机之间的分割符号插入到原来的位置.

$$A' = 365018079024$$

$$B' = 920875064031$$

(6) 变异方法和终止条件

实施过程中采用的变异概率为 8%.当变异条件激发后,采用的变异方法是:在被选中的染色体随机选取两台计算机,然后在两台计算机中随机选择两个任务进行交换,从而使解能够全局化.

终止条件根据遗传算法收敛条件和迭代次数来确定.遗传算法如果收敛,则迭代终止,这里采用的收敛条件是 $E_i < m$, m 是计算机的台数;如果遗传算法不收敛,当迭代 1 000 次后,遗传算法停止迭代.

3.3 负载平衡处理

为了加快遗传算法的收敛速度,对群体中的染

色体进行适当的负载平衡处理^[8].处理策略为:

- (1) 在染色体中找出计算机负载最大、最小的计算机.
- (2) 在负载最大的计算机中随机选择一个任务.
- (3) 如果负载大的计算机减去这个任务的负载,并且负载小的计算机加上这个任务后,若负载小的计算机比负载大的计算机负载依然小,则将这个任务从负载大的计算机中删除,并且将其插入到负载小的计算机中.
- (4) 更新两台计算机的临时负载.
- (5) 循环执行这样的负载平衡处理 k 次.
- (6) 逐个对其他染色体进行同样的负载平衡处理.

对一个染色体来说,负载平衡处理的次数增加,一方面会使得染色体越来越向最优解逼近,从而会使得遗传算法的收敛速度加快,减少迭代次数另一方面,负载平衡处理次数的增加,会增加平衡处理的时间,如果用 Sum 来表示负载平衡处理的时间复杂

度,则 Sum 由式(8)可以得到,其中 k 表示负载平衡处理的次数,generation 表示遗传算法迭代的次数,chrom 表示群体中染色体的个数, m 表示计算机台数.

Sum = O($m * chrom * k * generation$) (10)

4 仿真结果与分析

为了考察本文提出的调度策略和传统算法的执行效率,现进行一系列的对比.首先将任务在一台计算机上执行,然后将任务映射给异构环境下的 8 台计算机,其中任务数量变化范围为 512~4 096,任务粒度是随机生成,最大粒度任务的执行时间为 10min.负载平衡处理对遗传算法的影响如图 1 所示.在实验中,任务的 CPU 时间为 0 到 100 的随机数,内存为 0 到 900 的随机数.在代表程序运行之前,最大等待时间以 70 步长的时间随机递增,以提供一系列的任务.节点 CPU 最大时间为 1 000 到 3 000 的随机数,总体内存为 4 000 到 12 000 的随机数.调度算法性能对比如表 1 所示.

表 1 调度算法性能对比表

任务总数			服务器数目	总任务时间			丢弃任务								
高 QoS	中 QoS	低 QoS		Min-Min 算法	传统遗传算法	混合调度算法	Min-Min 算法			传统遗传算法			混合调度算法		
							高	中	低	高	中	低	高	中	低
149	40	40	8	856.55	845.32	745.34	0	1	0	0	1	0	0	0	0
159	35	39	8	834.63	832.44	711.34	0	1	3	0	1	2	0	0	1
138	44	46	8	963.31	944.21	832.98	0	2	2	0	2	1	0	1	0
150	41	38	8	1 098.26	987.98	887.35	0	3	2	0	2	1	0	1	0
154	40	35	8	859.15	843.31	753.45	0	1	1	0	0	0	0	0	0
157	40	34	8	942.44	931.12	837.01	0	3	2	0	2	2	0	1	0
155	40	35	8	869.12	845.43	765.45	0	2	2	0	2	1	0	1	0
156	40	44	8	1 096.59	985.34	768.45	0	1	1	0	0	1	0	0	1
151	40	39	8	755.38	752.29	642.32	0	2	1	0	1	0	0	0	0
159	39	32	8	801.57	799.26	643.05	0	0	2	0	0	1	0	0	0

图 1 给出了在迭代 1 000 次时,使用负载平衡处理和不使用负载平衡处理时对遗传算法收敛的影响,从图 1 可以看出,当遗传算法使用了负载平衡处理时,最大、最小负载差的值基本上收敛,而没有采用负载平衡处理时,在迭代 1 000 次时,遗传算法基本上发散,所以负载平衡处理可以极大地提高遗传算法的收敛速度.

从图 1 可以看出,改进的遗传算法相对于传统的算法在运行时间上要少,在性能上有明显的提高,而且在任务的丢弃方面,本文提出的算法任务丢弃

数量比其他的算法任务丢弃少,从而表明本文提出的算法比其他算法有着更优秀的执行效率.

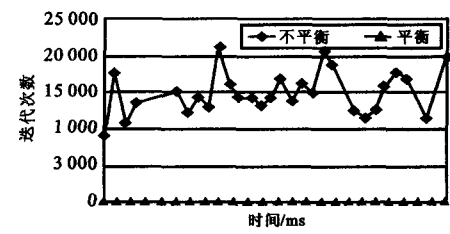


图 1 负载平衡处理对遗传算法的影响

5 结束语

本文在研究了近几年异构环境下的任务调度算法之后,提出了一种基于改进遗传算法的混合调度策略,此调度策略首先将任务进行分组,然后对每一组任务用改进的遗传算法来进行调度处理,在调度过程中,使用负载均衡方法来调整染色体中任务的分布.实验表明,本文提出的任务调度方法能有效减少平均任务执行时间,提高资源利用率,具有较好的性能.

参考文献:

- [1] 万本庭,陈明,鲁强.一种 QoS Min-Min 异构分布式系统任务调度策略[J]. 计算机工程, 2007, 33(11): 45-48.
- [2] Foster I, Kesselman C. The grid2 blueprint for a new computing infrastructure [M]. San Francisco: Morgan Kaufmann Publishers, 1998: 104-106.
- [3] Abraham A, Buyya R, Nath B. Nature's heuristics for scheduling jobs on computational grids[C]// The 8th IEEE International Conference on Advanced Computing and Communication. India, 2000: 121-123.
- [4] Braun T, Siegel H, Beck N. A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems[J]. Journal of Parallel and Distributed Computing, 2001, 61(6): 58-61.
- [5] 陈明. 分布式应用模型[M]. 北京: 科学出版社, 2009.
- [6] 王小平, 曹立明. 遗传算法[M]. 西安: 西安交通大学出版社, 2002.
- [7] 邱卫东. 一种实时异构嵌入式系统的任务调度算法[J]. 软件学报, 2004(4): 504-511.
- [8] Kwok Y K, Ahmad I. Dynamic critical-path scheduling: an effective technique for allocating task graphs to multiprocessors[J]. IEEE Transactions on Parallel and Distributed Systems, 1996, 7(5): 65-68.

作者简介:

范敏 男, (1967-), 硕士, 副教授. 研究方向为并行计算.
胡伟 男, (1979-), 博士研究生. 研究方向为并行复杂网络算法.

(上接第 118 页)

参考文献:

- [1] Shin C, Kim S, An S. Stable gateway selection scheme based on MANET with Internet[C]//The Sixth IEEE International Conference on Computer and Information Technology. Seoul, USA, 2006: 80-80.
- [2] Jonsson U, Alriksson F, Larsson T, et al. MIPMANET: mobile IP for mobile Ad hoc networks[C]//2000 First Annual Workshop on Mobile and Ad Hoc Networking and Computing. Boston, USA, 2000: 75-85.
- [3] Sun Y, Belding-Royer E M, Perkins C E. Internet connectivity for Ad hoc mobile networks[J]. Int. Journal of Wireless Information Networks, 2002, 9(2): 465-482.
- [4] 宋文, 方旭明. 无线 Mesh 网络公平感知路由算法设计与仿真[J]. 系统仿真学报, 2007, 19(18): 4320-4325.
- [5] Toh C K. Associativity-based routing for Ad-hoc mobile networks[J]. Wireless Personal Communications Journal, 1997, 4(2): 3-10.
- [6] 张凯捷, 向勇, 史美林. 移动自组网与 Internet 的多网关互连[J]. 清华大学学报: 自然科学版, 2007, 47(1): 100-103.
- [7] Li Taoshen, Chen Songqiao, Yan Chen, et al. Research on anycast routing algorithm with multiple QoS parameters constraint[J]. Journal of Communication and Computer, 2005, 2(4): 54-60.

作者简介:

丘小兰 女, (1984-), 硕士研究生, CCF 学生会员. 研究方向为无线 Mesh 网络.
李陶深 男, (1957-), 博士, 教授, CCF 高级会员. 研究方向为无线 Mesh 网络、网络计算与路由、分布式数据库等.
葛志辉 男, (1980-), 博士, 副教授. 研究方向为无线 Mesh 网络性能优化、路由协议设计.