一种基于改进遗传算法的网格任务调度策略①

An Improved Genetic Algorithm-Based Task Scheduling Strategy in Grid

张 阳 黄文明 兰 静 (桂林电子科技大学 计算机与控制学院 广西 桂林 541004)

摘 要: 针对网格环境下任务的调度问题,本文提出了一种改进的快速收敛的遗传算法.通过调整算法结构,增加了对染色体的分割与重组操作.使遗传算法能快速收敛.仿真实验表明,与标准调度算法相比,提出的算法取得了很好收敛速度。

关键词: 网格 任务调度 改进遗传算法 分割

网格[1]就是一个集成为一台巨大的超级计算机,或者说是一个计算资源池[2],它可以实现全球范围的计算资源、存储资源、数据资源、信息资源、知识资源、专家资源、设备资源,甚至是人才资源等各种相关的广泛分布的资源的全面共享。然而,网格系统呈现的性能因为资源竞争和任务不确定性等因素而高度动态变化。因此网格任务管理是计算网格中一个关键性的研究课题。

1 引言

本文针对传统的遗传算法搜索空间大、计算量重,特别是当搜索具有复杂染色体结构的求解空间时,收敛速度很慢的问题,提出了一种改进的相对快速收敛的GA算法,并使之可以应用到网格任务调度管理.此改进算法[3]根据遗传算法基本原理,增加了对染色体的分割(dividing)与重组(recombination)操作,该算法首先依据于各段的结构和段长,分别初始化随机生成的N个染色体段(chromosome segment)组成的段群体(segmented population),然后对各段群体独自实施遗传操作以寻找优化段,最后再对选出的优化段实施重组操作,重新组合成完整的染色体来搜索优化解。

2 网格计算任务调度

网格任务的调度属大规模的资源受限的项目调度问题RCPSP (Resource-Constrained Project Scheduling Problem)。因此,如何提高搜索性能,成为人们关注的问题。

任务调度首先根据任务的需求,发现满足条件的计算资源;然后从满足条件的计算资源中根据主要因素或选择策略选择一个合适的资源分配给该任务:任务获得满足条件的资源后,可以在该资源上运行,并处于资源本地的任务管理机制的管理之下;任务在资源上执行结束后,把占用的资源还给网格管理机构,网格任务管理模块把任务执行结果和有关信息告诉任务提交者。

在网格环境下,通常考虑的是一组相互独立,彼此之间没有通讯和数据依赖的元任务的调度。一个异构系统由m台异构机器组成,有n个独立任务要竞争使用机器,分配的目标是把这n个任务合理地分配到m台机器上执行,使总的执行时间最短。用一个n*m矩阵ETC来表示任务的估计执行时间,其中元素ETC(i,j)表示任务ti在机器pj上的估计执行时间。定义机器运行时间Tj为机器完成分配给它的所有任务所需的时间,任务ti在机器pj上的完成时间ctij=Tj + ETC(i,j).任务ti的最小完成时间MCTi是所有ctij中的最小值。任务分配的目标是使最大运行时间的机器的运行时间最短:

$$\min\left(\max_{j=1}^{m}(T_j)\right) \quad (1)$$

① 基金项目:广西研究生创新计划资助项目(2008105950812M428);广西教育厅项目(2004(20));桂林电子科技大学06年度学科软环境项目
收稿时间:2008-10-16

可见,网格计算的任务调度是一个 NP 完全问题[4],至今没有找到可以精确求得最优解的多项式时间算法,由于指数型算法对解决规模较大的问题又十分困难,以求近似最优解为目的的启发性算法自然就受到了极大的重视[5]。

3 遗传算法

遗传算法(GA, Genetic Algorithm)是 Holland 于 1975 年提出的一种借鉴生物界自然选择和进化机制发展起来的高度并行、随机、自适应搜索算法。遗传算法是一种基于达尔文的进化论与孟德尔和摩根的遗传学理论的全局性搜索算法[6].由于其思想简单、易于实现以及表现出来的鲁棒性,已经得到了许多应用,在优化和搜索、智能控制、模式识别以及人工生命等领域均获得了令人鼓舞的成就。

任务调度是一个 NP 完全问题,因此研究在搜索过程中自动获得和积累有关搜索空间知识并自适应地完成搜索过程,从而得到最优解或次优解的通用的搜索算法一直是令人瞩目的课题。遗传算法就是这种特别有效的算法。

3.1 遗传算法基本原理

遗传算法从一组随机产生的初始解称为“种群(Population)”开始搜索过程。种群中的每个个体是问题的一个解,称为“染色体(Chromosome)"。在遗传算法中最重要的概念是染色体,染色体通常是一串数据(或数组),用来作为优化问题的解的代码,其本身不一定是解。这些染色体在后续迭代中不断进化,称为遗传。在每一代中用“适应度函数(Fitness)”来测量染色体的好坏。生成的下一代染色体称为后代(Offspring),后代是由前一代染色体通过交叉(Crossover)或者变异(Mutation)运算形成的。新一代形成中,根据适值的大小选择(Reproduction)部分后代,淘汰部分后代,从而保持种群大小是常数,适值高的染色体被选中的概率较高。据此,经过若干代之后,算法收敛于最好的染色体,它很可能就是问题的最优解或次优解。

标准遗传算法的流程图如图 1 所示。

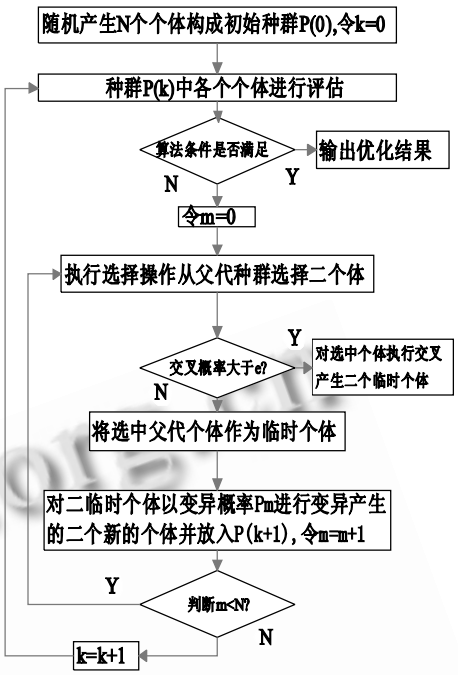

图 1　标准遗传算法流程图

3.2 遗传算法的优点和缺点

遗传算法在本质上是一种概率搜索算法.并行性和全局解空间搜索是 GA 的两个最显著的特点。遗传算法具有十分顽强的鲁棒性,主要有以下优点[7]:

① 遗传算法的并行性。遗传算法按并行方式对种群进行搜索,而不是按单点进行搜索。它的并行性可让几百甚至数千台计算机进行独立种群的演化计算,待所有机器都运算结束时再进行比较,选取最佳个体。另外由于采用种群的方式搜索,可同时搜索解空间内的多个区域,并相互交流信息。

② 遗传算法不需要其他辅助知识,只需要影响搜索的目标函数和适应度函数。

③ 遗传算法的处理对象不是参数本身,而是对参数集进行了编码的个体。对此编码操作,使得遗传算法可直接对结构对象进行操作。所谓结构对象泛指集合、序列、矩阵、树、图、链和表等各种一维或二维甚至三维结构形式的对象。这一特点,使得遗传算法具有广泛的应用领域。

但是,遗传算法作为一种优化方法,存在自身的局限性:

① 单一的遗传算法编码不能全面地将优化问题的约束表示出来,这样,计算的时间必然增加。

② 遗传算法本身参数的选择缺乏定量的标准目前采用的基本上是经验值而且不同的编码方法,通过的遗传技术都会影响到算法参数的选择,因而会影响到算法的通用性。

③ 遗传算法容易出现早熟收敛。它主要表现在两

个方面：一方面群体中所有的个体都陷于同一极值而停止进化.另一方面接近最优解的个体总是被淘汰，进化过程不收敛。

4　改进的遗传算法设计

4.1 染色体编码

本文采用间接编码方式,假设任务数为 p,机器数为 q。染色体上的每个基因的位置编号代表任务编号,染色体长度等于 p,每个基因位用{0,1,…,q-1}之间的整数表示,代表该任务占用的机器编号。例如,机器数 q 为 3,任务数 p 为 10,则编码(0,1,1,0,1,2,0,1,0,2)表示第 1、4、7、9 个任务分配在第 1 台机器上,第 2、3、5、8 个任务分配在第 2 台机器上,第 6、10 个任务分配在第 3 台机器上。针对本文的讨论的问题,某个任务在特定机器上的估计运行时间是一定的,机器的 Makespan 与任务在机器上的执行次序无关,所以可以采用这种简单的编码方式,不仅能使算法容易实现,而且提高了算法的执行效率。

4.2 初始种群的产生

按照染色体的编码要求,随机产生规模为 80 的种群

4.3 适应度函数

适应度用来评价染色体的优劣，本文采用界限构造法,设置当代种群中最差个体的调度长度为 Cmax,则适应度函数为:

$$F(C)=Fitness(q_j)=C_{max}-Makespan(q_j)+r \quad (2)$$

其中，r 为调节因子

4.4 染色体分割

对染色体 C 实施分割操作，生成 k 个染色体段。基于每个染色体段各自随机生成 k 个段群体,PUi={C_{ij}},其中 i(i=1,2,…,K)为段群体号,j(j=1,2,…,N)为段群体内染色体号。

4.5 段适应度函数

$$f^{g-1}(C_{ij}^{g+1}) = F(C_1^{g+1}) = F(C_{11}^{g+1} \cup C_{21}^{g+1} \cup ... \cup C_{i-1,1}^{g+1} \cup C_{ij}^{g} \cup C_{i+1,1}^{g} \cup ... \cup C_{k1}^{g}) \quad (3)$$

其中，上标 g(g≥1)表示代数，C_{i1}^{g} (i=1,2,…,k)是依据段适应度函数 $f^{g}(C_{ij}^{g})$(j=1,2,…,N)求得的第 g 代的各段群体中的最优染色体段,第 g+1 代的各段群体中的各染色体段的适应度值由 $f^{g+1}(C_{ij}^{g+1})$求得,对第 i 段群体求其 g+1 代最优染色体段 C_{i1}^{g+1} 要用到 g+1 代最优染色体段(当段号小于 i 时)和 g 代最优染色体段(当段号大于 i 时),依此式求得的 $f^{g+1}(C_{ij}^{g+1})$ 越大,表示段 C_{ij}^{g+1} 越优,最后在 N 个染色体段中选取一个最优的命名为 C_{i1}^{g+1}。由(3)式可知 f^{g+1} 随 g 代和 g+1 代各段群体中的最优染色体段的不同而不同。依据(3)式计算各 PU_i 中染色体的段适应度 $f^{g+1}(C_{ij}^{g+1})$,并将段群体内的染色体分别按其段适应度值由大到小排序。

4.6 遗传搜索

搜索过程如下:

```
g=0;
for(  ;  ;){
        g=g+1;
   for(i=1;i<=k;i++)
```

{对段群体 PUi 按传统的遗传算法实施遗传操作产生 2m 个子染色体,2m<<N;

替换 PUi 中段适应度值小的 2m 个染色体,即 PUi 中的后 2m 染色体;

```
   for(j=1;j<=N;j++)
   {
```

$C_1^{g+1} = C_{11}^{g+1} \cup C_{21}^{g+1} \cup ... \cup C_{i-1,1}^{g+1} \cup C_{ij}^{g} \cup C_{i+1,1}^{g} \cup ... \cup C_{k1}^{g}$

计算 $F(C_1^{g+1})$;

if(C_1^{g+1} 满足目标解)

解码 C_1^{g+1},得到目标解,算法结束;

else

$f^{g+1}(C_{ij}^{g+1}) = F(C_1^{g+1})$;

}

按 PUi 中段适应度值由大到小将染色体重新排序;

}

}

假定一次遗传操作产生 2n 个子染色体,那么传统的 GA 需要计算适应度的次数为 2n 次;改进的算法对于一个段群体的一次遗传操作若产生 2n 个子染色体,则实际上对于 K 个段群体来说相当于产生了 $2n \cdot N^{k-1} \cdot k$ 个子染色体,而对于适应度的计算量为 $K \cdot N$ 次;对于传统的 GA 来说若产生 $2m \cdot N^{k-1} \cdot k$ 个子染色体,则计算子染色体适应度的量亦为 $2m \cdot N^{k-1} \cdot k$ 次,显然下式成立:

$$2m*N^{k-1}*k>k*N \quad (4)$$

显然因而改进的算法在产生同样个数的子染色体时能大大减少其适应度的计算量,故能大幅度加快收敛过程。

5　实验结果

在 GridSim[8]仿真平台中，取机器数为 20,每个机器的设置参数如下:

表 1　每个机器的参数

处理器个数(n)	处理能(MIPS)	通信能力(Mb/s)
2	337	100

设定每个 DAG 图的任务数大约为 100 个，每个任务的平均长度大概 1000MI。

遗传算法的参数设置如下

表 2　遗传算法的参数

种群大(Pop_Size)	交叉概率(Pc)	变异概率(Pm)
80	0.5	0.01

使用传统遗传算法和改进算法适应度值变化如下二图所示:

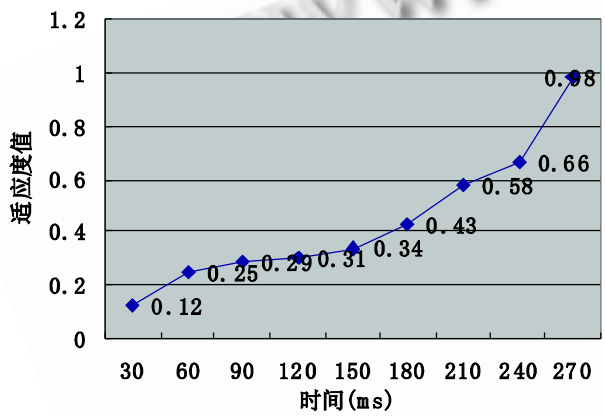

图 2　传统 GA 的适应度变化曲线

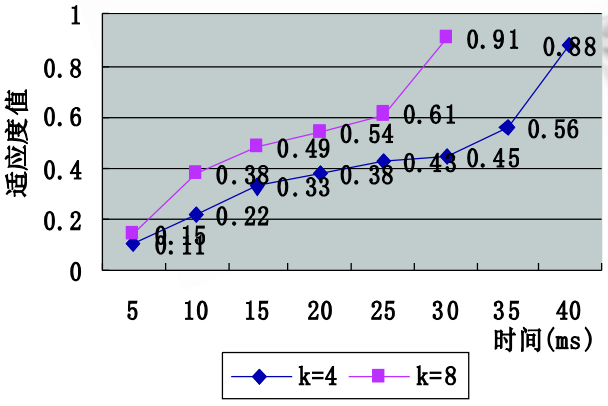

图 3　用改进的快速 GA 的适应度变化曲线

图 2 为使用传统遗传算法(即 K=1 的情况)解决网格任务问题的适应度随时间变化曲线,图 3 为使用快速收敛的遗传算法时的适应度随时间变化曲线,其中图 3 给出了 K=4,8 的二种情况。从两图的对比可以看出使用改进的快速收敛遗传算法,在较短的时间内适应度值迅速增加,从而极大加快了收敛速度。

6　总结

针对网格环境下的任务调度问题,人们提出了很多算法。本文提出了一种改进的相对快速收敛的 GA 算法，增加对染色体的分割与重组操作,依据于各段的结构和段长,组成段群体,对其实施遗传操作以寻找优化段,重新组合成完整的染色体来搜索优化解，段重新组合后又不致于减少搜索空间的大小,因此,该算法极大加快了收敛速度。仿真实验表明,本文的算法的收敛性能明显高于传统遗传算法。下一步研究的方向是根据网格的特点将针对早熟收敛等问题进行研究。

参考文献

1 都志辉,陈渝,刘鹏.网格计算.北京:清华大学出版社, 2002:204－211.

2 The DOE Common Component Architecture Project http://www.extreme.indiana.edu/~gannon/cca_report.html.

3 周春光.一种快速收敛的遗传算法.软件学报, 1996,7(增刊):311－314.

4 Markel J D. FFT Pruning.IEEE Transactions on Audio and Electroacoustics, 1971,19(4):305－311.

5 刘家壮,徐源.网络最优化.北京:高等教育出版社, 1991.

6 Tu YK, Yang J, Sun MT, et al. Fast variablesize blockmotion estimation for efficient H.264/AVC encoding. Signal Processing: Image Communication, 2005,20(7):595－623.

7 王晓平,曹立明.遗传算法一理论、应用于软件实现.西安:西安交通大学出版社, 2002.

8 刘一萌.基于 GridSim 的网格资源调度算法研究[硕士学位论文].成都:四川大学, 2004.