基于改进遗传算法的双层网格任务调度①

A Double-Deck Grid Task Scheduling Based on Improved Genetic Algorithm

陈友文[1,2]　文雄军[1]　(1.湖南涉外经济学院 计算机学部 湖南 长沙 410205;
2.湖南大学 计算机与通信学院 湖南 长沙 410082)

摘　要： 合理的任务调度算法可以在很大程度上提高网格系统的利用率。针对网格环境异构、分布等特点，提出了基于改进遗传算法(MRNGA)的双层网格任务调度算法，在简单遗传算法的基础上改进选择算子并引入了小生境技术，对网格结构采用双层编码的方式，仿真实验表明，该方法具有良好的搜索能力和资源负载均衡度，对异构系统中的任务调度具有较好的处理结果。

关键词： 网格 任务调度 改进遗传算法 选择算子 小生境

网格计算(Grid Computing) 是当前互联网研究中的一个热点，也是并行和分布处理技术的一个发展方向。在网格计算中,任务调度是其重要的组成部分，网格任务调度就是按照一定的规则，将用户提交给网格环境中的任务分配给资源中的各个节点的过程。合理的任务调度算法可以在很大程度上提高网格系统的有效利用率，并且网格任务调度已被证明是一个 NP 完全问题[1]。因此，网格任务调度算法具有很高的研究价值。

在以往的研究中产生了许多网格任务调度算法，如 Min-Min 和 Min-Max 算法、Sufferage 算法和遗传算法等。Min-Min 和 Min-Max 算法和 Sufferage 算法实现简单，有较小的 Makespan 值，但是负载平衡性能都很差。用遗传算法进行网格任务调度，有很好的 Makespan，负载平衡性也很好[2]。但遗传算法的随机性，使算法有可能失效，而且可能会陷入局部最优，鉴于此，本文提出了一种改进的遗传算法(MRNGA)来进行网格任务调度。

1　调度模型

任务之间的关系用一个加权的有向无环图 G 来表示，公式为 G=(V,E,R,W,C)其中:

V: m 个子任务的集合;

E: 子任务之间有向边的集合，有向边(V_i,V_j)(其中$V_i,V_j \in V$)表示任务v_j在任务v_i完成之后才能开始执行;

R: n 个计算资源的集合,一般 n<m;

W: 表示计算开销，是一个 $m \times n$ 的矩阵,W_{ij} 表示任务 v_i 在资源 r_j 上的执行时间;第 i 行表是第 i 个任务 v_i 分别在 n 个资源上的调度时间。

C: 表示通信开销，是一个 $m \times m$ 矩阵,表示任务v_i和任务v_j之间的通信延迟,如果这两个任务分配到同一个资源上,则认为通信开销为 0；如果存在一条从 v_i 到 v_j 的路,称 v_i 是 v_j 的前驱节点,而对于$(V_i,V_j) \in E$,称 v_i 是 v_j 的立即前驱节点,记为 $V_i \in iPred(V_j)$,称 v_j 是 v_i 的立即后继节点,记为 $V_j \in iSucc(V_i)$。

2　算法描述

2.1 改进遗传算法(MRNGA)思想

遗传算法是美国 Michigan 大学 J.Holland 教授于 1975 年提出的,以达尔文生物进化理论“适者生存,优胜劣汰”为基础,模拟生物进化过程。GA 具有许多明显的优势,如:具有领域无关的群体性全局搜索能力;使用评价函数启发搜索,过程简单;易于与其它优化技术或已有模型结合,可扩展性强[3],应用在任务调度算

① 基金项目:湖南省教育厅科研资助项目(08C515)
收稿时间:2009-03-31

法中具有其独有的优势。

然而算法由于随机操作的原因,往往会产生较大的随机误差,有时甚至会出现“退化”现象,即适应度较高的个体失去选择机会。为了减少随机性的误差，本文在轮盘赌的基础上，提出了一种多轮轮盘赌的选择算子。为了改善遗传算法固有的早熟现象，本文还引入了小生境技术，小生境的最优保留和排挤策略使得算法在保证多样性的同时能够保留最优解。

2.1.1 改进的多轮轮盘赌选择算子

令 M 为种群大小,每个个体 i 的适应度为 F_i,根据这 M 个个体的适应度计算出的选择概率将[0,1]划分为 M 个区间, $\xi_1, \xi_2, \ldots, \xi_M$ 为 M 个整数, 分别代表这 M 个区间所落的随机数的个数. 算法进行多轮轮盘赌选择. 在多轮轮盘赌选择过程中,利用产生的 M 个随机数进行一轮选择, 统计各区间的 ξ 值,得到 $\xi_1, \xi_2, \ldots, \xi_M$, 取最大 ξ 值所在区间对应的个体为本轮所选中的个体,重复以上操作 M 次,从而得到 M 个个体。

算法详细过程如下:

设父代种群 $Z=\{a_1,a_2,\ldots,a_i,\ldots,a_M\}$，其中每个个体的适应度大小为 F_{a_i} ，子代群体初始状态设为 $Y=\Phi$ ，则该选择算子的具体执行过程为:

(1) 计算出群体 Z 中所有个体的适应度的总和

$$\sum_{i=1}^{M} F_{a_i} \text{ , (i=1,2,3,...,M).}$$

(2) 计算出每个个体被选取的概率

$$P_{a_i} = F_{a_i} / \sum_{i=1}^{M} F_{a_i} \text{ , (i = 1, 2, 3, . . . ,M).}$$

(3) 转动 M 轮

① 产生 M 个[0,1]之间的随机数.

② 统计每个区间的 ξ 值 $\xi_1, \xi_2, \ldots, \xi_M$,其中 ξ_i 是落在区间 i 上的随机数个数,i=1,2,...,M。

③ 取最大的 ξ 值， $\xi_j=\max\{\xi_1, \xi_2, \ldots, \xi_M\}$, 所在区间 j 对应的个体 a_j 作为本轮选择的个体 y_i。

④ 将 y_i 并如子代群体 Y 中,即

$$Y=\begin{cases} Y(0)=\Phi \\ Y(t)=Y(t-1)\cup y_i \end{cases}$$

⑤ 若选出的个体数达到了种群的大小 M,则转 4),否则转①。

(4)存储所有选择的个体,返回.

本文提出的改进的多轮轮盘选择算子将传统轮盘赌算子中的每产生一个随机数就确定一个个体改进为每产生 M 个随机数才能确定一个个体,增大了产生随机数的数量,由原来的 M 增至 M2,这样能更精确的体现随机数的作用,以便误差的减小。

2.1.2 与小生境技术的结合

GA 在早期进行粗略搜索时,容易丢失最优解,在后期进行精细搜索时容易陷入局部最优解。为了解决这些问题,将小生境技术引入到算法中。小生境技术的最优保留原则使得算法在保证多样性的同时能够保留最优解;小生境技术通过海明距离定义的排挤策略能够在算法的后期仍然维持较高的多样性。这种改进了选择算子和基于小生境的算法,就是本文提出的一种新的遗传算法(MRNGA)。

2.2 染色体编码

本算法采用了文献[4]的双层染色体进化编码，结构见表 1。

表 1　染色体结构

t_1	t_2	t_3	…	t_m
r_1	r_2	r_3	…	r_m

由表 1 可见，染色体上层为任务层，其中，t_i 代表第 i 个被调度的任务，其值为任务的编号；下层为网格资源层，其中，r_i 代表 t_i 所分配的网格资源，其值为网格资源的编号；行方向表示了任务的调度顺序；列方向(t_i, r_i)为任务-网格资源对，体现了 t_i 与 r_i 的对应关系，二者构成一个基因。

2.3 初始种群(initial population)

算法首先生成一个初始种群，然后在此基础上执行各种操作。

2.3.1 高度分层排序

染色体基因顺序必须满足对应的任务先后依赖关系，否则为无效染色体。为保证遗传操作后染色体的有效性，对图中各个任务按高度分层排序[5]，其计算公式如下:

$$Height(v_i)=\begin{cases} 0 & if ipred(v_i)=\Phi \\ \max\limits_{v_j \in ipred(v_i)} Height(v_j)+1 & \text{其它} \end{cases}$$

其中，Height(v_i)代表节点 v_i 的高度。高度相同的节点归为一层。通过分层，明确了任务的先后调度关系，保证遗传操作后的染色体的有效性[6]。

2.3.2 初始种群生成算法

初始种群生成算法步骤如下：

① 计算任务图 G 中每个任务的高度 Height；

② 将 G 中所有任务按高度划分为不同的子集 SG(h)，即高度为 h 的任务集；

③ 所有任务按任务集高度升序排列，同任务集的各任务按随机顺序排列，生成染色体的任务层序列；

④ 将每个任务 v_i 按步骤③中得到的顺序，在矩阵 W 中找出该任务 v_i 对应的第 i 行中最小的元素 $Min(v_{ij})$，其中 $i \in [1,m], j \in [1,n]$，即对任务 v_i 调度时间最短的资源 r_j，作为染色体的资源层。这样就保证了较小的 Makespan。

⑤ 步骤③、步骤④生成一个染色体，按照设定的种群大小 pSize，重复步骤③、步骤④pSize 次，生成初始种群。

2.4 适应度函数

算法使用适应度函数来评价染色体优劣。在本算法中，适应度函数取决于调度的完成时间。适应度函数为

$$Fitness(c_i)=\frac{1}{FT(c_i)}$$

其中，$Fitness(c_i)$表示种群中第 i 个染色体 c_i 的适应度；$FT(c_i)$为 c_i 的调度完成时间。

2.5 遗传操作

初始种群生成后即可对种群中代表调度序列的染色体进行遗传操作。遗传操作包括选择、交叉和变异 3 种基本形式。

(1)实现选择操作，采用上面提到的多轮轮盘赌选择算子。根据选择概率，适应度较高的个体容易被保留下来。多轮轮盘选择算子将传统轮盘赌算子中的每产生一个随机数就确定一个个体改进为每产生 M 个随机数才能确定一个个体，增大了产生随机数的数量，由原来的 M 增至 M^2，这样能更精确的体现随机数的作用，以便误差的减小。

(2)交叉操作，随机选择某一高度，在被选中进行交叉操作的 2 个个体中，将属于该高度集序列的最左边的基因作为的交叉点，在其交叉点左边的基因保持不变，交叉点及交叉点以右的基因进行互换。

(3)变异操作，主要步骤如下：

① 随机选取任一基因及与该基因任务高度相同的另一基因；

② 如果 2 个基因的网格资源相同，则交换 2 个基因的任务产生新个体；否则，交换 2 个基因的网格资源，产生新个体。

执行完一次遗传操作之后，重新计算个体的适应度。

2.6 小生境淘汰运算

按照下式求出中每两个个体 b_i 和 b_j 之间的海明距离：

$$\left\|b_i-b_j\right\|=\sum_{k=1}^{chromlen}\left(b_{i_k}-b_{j_k}\right)^2$$

(i=1,...,M-1； j=i+1,...,M； M 为种群的大小；Chromlen 为染色体长度)

当 $\left\|b_i-b_j\right\|<L$ (L 为设置的最小海明距离)时，比较个体 b_i 和个体 b_j 的适应度大小，并对其中适应度比较低的个体处以罚函数：$\min\left(F_{b_i},F_{b_j}\right)=Penalty$，其中 Penalty 为一个很小的正数。

2.7 算法遗传迭代终止条件

本算法采用如下遗传迭代终止条件：

(1) 设定最大遗传迭代次数 MAXloop，当迭代次数超过 MAXloop，迭代终止；

(2) 设定终止代数 ENDgen 和终止阈值 ENDthr，在设定的迭代次数范围内，如果连 ENDgen 代，子代种群最优解的变化值都小于 ENDthr，说明算法收敛到最终解，该解即为具有最短完成时间的调度方案，算法终止。

2.8 实验结果及分析

本文对任务调度算法进行了仿真实验，以验证算法的正确性并对其性能进行评价.实验中，模拟了由三个站点组成的网格计算环境下的网格任务调度情况。算法主要参数为：种群大小为 150，杂交概率为 0.9，变异概率为 0.1，实验结果见图 1、图 2。

图 1 表示的是 MRNGA、GA、Min-Min[7]算法在不同任务数情况下获得最终解所需时间比较图，可以看出，任务节较少时，3 种算法运行时间相差不多，随着任务数增多，MRNGA 算法求解速度快，时间明显少于其他两种算法。图 2 给出了 3 种算法的负载均衡性比较，其中，Min-Min 算法中每个资源的负载极不平衡，主要原因在于该算法将执行时间最短的任务分配在负载最小的节点上，这导致执行时间长的任务分配到负载较大的节点上。MRNGA 算法的曲线最平滑，资源负载均衡度最接近 1，表明该算法的负载均衡性高。

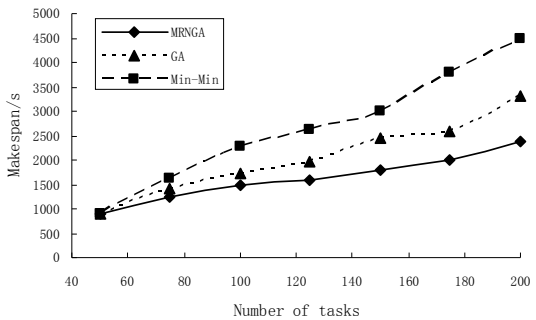

图 1 MRNGA、GA、Min-Min 算法在不同任务数情况下的时间跨度比较

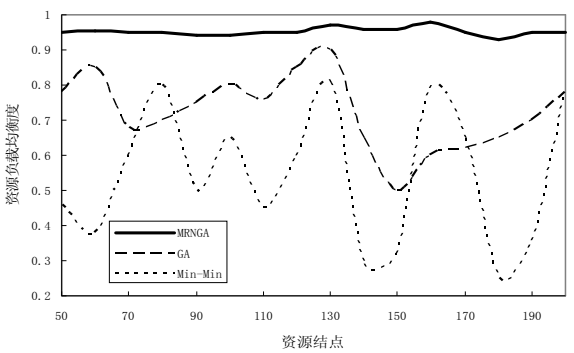

图 2 MRNGA、GA、Min-Min 算法的资源负载均衡度比较

3 结论

本文提出了一种基于改进基本遗传算法的选择算法并引入小生境技术(MRNGA)的双层网格任务调度算法。本算法通过合适的双层编码方式以及合适的调度策略，缩短了调度时间，提高了资源调度性能。实验表明，本算法具有良好的调度结果与求解速度，是一种有效的网格任务调度方法。

参考文献

1 陈锋,刘宗田,石振国,王莉.基于禁忌搜索算法的网格任务调度.计算机工程, 2007,33(21):75－77.

2 Braun TD, Siegel HS, Beck N. A Comparison of Eleven Static Heuristics for Mapping a Class of Independent Tasks onto Heterogeneous Distributed Computing Systems. Journal of Parallel and Distributed Computing, 2001,61(6):810－837.

3 王小平,曹立明.遗传算法理论应用与软件实现.西安:西安交通大学出版社, 2006:46－52.

4 杨博,陈志刚,刘立.基于融合进化计算的网格任务调度算法.计算机工程, 2007,33(18):181－183.

5 Ratnaweera A, Halgamuge SK, Watson HC. Self-organizing Hierarchical Particle Swarm Optimizer with time-varying Acceleration Coefficients. IEEE Transactions on Evolutionary Computation, 2004,8(3):240－255.

6 吴雄奇,曾文华.基于改进遗传算法的网格资源调度算法.微电子学与计算机, 2006,23(9):26－29.

7 Sih GC, Lee EA. A compile-time Scheduling Heuristic for Interconnection-constrained Heterogeneous Processor Architectures. IEEE Transactions on Parallel and Distributed Systems, 1993,4(2):308－323.