改进遗传算法在网格任务调度中的应用

卜艳萍
（上海交通大学 技术学院，上海 201101）

摘 要：为了提高遗传算法的搜索性能，同时满足网格资源的优化分配，提出了一种带过滤机制的遗传算法，使其适用于网格任务调度问题的优化处理。仿真研究表明该算法更符合网格调度的复杂环境，能得到较短的任务执行时间和较好的负载均衡性。

关键词：遗传算法；网格；任务调度；完成时间
中图分类号：TP393 **文献标识码**：A **文章编号**：1674-7720(2010)18-0095-04

Application of improved genetic algorithm to grid task scheduling

BU Yan Ping
(School of Technology, Shanghai Jiaotong University, Shanghai 201101, China)

Abstract: To improve the searching performance for genetic algorithm, and optimize the grid resources allocation, an improved genetic algorithm with filtrating is presented to the optimization of grid task scheduling problems. By utilizing this algorithm, shorter execution time and better performance of load balancing can be gained via the simulation studies, especially in grid scheduling environments.

Key words: genetic algorithm; grid; task scheduling; makespan

网格任务调度的基本思想是把分布在不同地理位置上的计算资源、存储资源、通信资源、软件资源、信息资源和知识资源等通过 Internet 整合成一台巨大的超级计算机，实现各种资源的全面共享[1]。网格技术的关键是资源管理，即有效地分配和使用网格资源。

用户通过向网格系统提交计算任务来共享网格资源，网格调度程序再按照某种策略把这些任务分配给合适的资源[2]。高效的调度策略或算法可以充分利用网格系统的处理能力，从而提高应用程序的性能。目前在网格调度算法研究中，其目标主要是增加吞吐率和系统的使用率，实现经济系统和用户的约束条件，以实现在整个系统中网格应用任务的完成时间最短。

另外，在网格环境中，由于任务到达的随机性以及各节点处理能力上的差异，会造成某些节点分配的任务过重，而另外一些节点却是空闲的，即出现负载不均衡现象[3]。因此，考察调度算法的性能时，也应考虑负载均衡性问题。

遗传算法（GA）是建立一个调度的集合并从其中找出优化的调度，将这种特性遗传给下一代。遗传算法通过适应度函数交叉和重组得出最优的调度。这是一种迭代的算法，它的优点是在不断进化的过程中吸收系统发生改变，能够适应动态变化的网格系统。

本文将改进的遗传算法（IGA）用于网格任务调度，用 IGA 寻找满足完成所有任务时间最短的优化方案，仿真实验表明，该方案性能良好。

1 问题描述

网格是一个集成的计算与资源环境，网格技术作为一种高性能广域分布式计算模型，已经成为众多研究机构的研究热点。

在网格技术的众多问题中，网格计算中的任务调度在一般形式下是一个 NP 问题，没有最优解。在调度算法的高效性、资源的异构性以及资源分配决策的并行性和分布性等方面，传统的调度算法并不能很好地适应网格资源的特点。因此，如何对网格资源进行合理分配和管理，满足各种应用的服务需求，实现资源的优化利用，就成为该领域的研究关键。

网格任务调度根据任务间是否存在通信关系可以分为对相互间存在通信任务的任务组的调度和对相互独立的任务组的调度[4]。在算法实现中都假定资源的信息是可获取的。

网格任务调度的实质就是在一个由 m 个需要调度的任务、n 个可用的任务执行单元(主机或集群)、k 个数据存储单元构成的网格环境下，把 m 个任务 $T=\{t_1,t_2,\cdots,t_m\}$ 以合理的方式调度到 n 台主机的过程，目的是得到尽可能短的总完成时间(makespan)。把每个执行单元广义地当作一台主机来看待，把 k 个数据存储单元当成一个存储系统整体来看待。

m 个任务在 n 台不同主机上的预测执行时间 *EET* 是一个 $m\times n$ 的矩阵。矩阵中的每一行代表某一个任务在 n 台机器上的不同执行时间，每一列代表在同一台机器上的 m 个任务的不同执行时间。

通信开销矩阵 *CCT* 描述了网格环境下，不同“任务对”在各主机之间进行数据传输的通信开销。任务必须分配到一台主机上，所需要的数据也必须从它的父任务发送过来，父任务可能有多个。第一个任务没有通信开销，其后的所有任务都可能有通信开销，因此 *CCT* 是一个 $m\times m$ 的矩阵。

m 个任务在 n 台不同机器上的预测最短完成时间 *MCT* 是一个 $m\times n$ 的矩阵，$MCT(i,j)$ 除了包含 $EET(i,j)$ 外，还应考虑通信和等待通信的时间开销 $T(i,j)$，$T(i,j)=\max((CCT(i,v)+TW(i,k)),k=1,2,\cdots,i-1)$，$TW(i,k)$ 为第 i 个任务等待其父任务准备好数据的时间。

调度的目标是在考虑通信开销和给定代价及约束集合现状之下，任务集合总的完成时间最短。

负载均衡性可以用每个调度方案中各台主机的最长执行时间与最短执行时间的比值来衡量，该值越小，则该调度方案中各台主机的负载均衡性越好。

2 改进遗传算法在网格任务调度中的应用

2.1 改进遗传算法

遗传算法是 Holland 于 1975 年受生物进化论的启发而提出的。并行性和全局解空间搜索是遗传算法的两个最显著的特点[5]。

遗传算法求解问题的基本步骤为：

(1)定义适应度函数和相应的隶属度函数；

(2)确定算法的初值和初始种群；

(3)对种群进行选择、交叉、变异操作，产生新的种群；

(4)循环遗传操作，直到达到进化的最大代数。

适应度比例方法是目前遗传算法中最基本也是最常用的选择方法，即轮盘赌选择法。在该方法中，各个个体(染色体)的选择概率和其适应度值成比例，适应度高的个体被选中的可能性大。

交叉运算是遗传算法区别于其他进化算法的重要特征，在遗传算法中起关键作用，是产生新个体的主要方法。两点交叉是遗传算法中比较常用的交叉方法。在相互配对的两个个体编码串中随机设置两个交叉点，然后在两个交叉点之间进行基因交换，以产生新的个体。

在群体的所有个体中随机地确定基因座，以事先设定的变异概率来对这些基因座上的基因值进行变异。当进行变异操作时，所有基因座上的基因值的变异必须是合法的，并且必须在可选择的范围内[6]进行。

改进遗传算法的基本思想是对种群中染色体进行过滤。首先计算出种群中每一个个体所对应的适应度函数值及负载均衡度值，剔除负载均衡度值低的部分染色体，再从剩余的染色体中选出一些适应度高的个体，其数量等于剔除掉染色体的数量。使这些个体在种群中复制一次，以保持种群大小不变。这样，高适应度的染色体取代负载均衡度值不良的染色体，使得高适应度染色体在种群中所占比例增大，从而使选择操作中有更多的优良个体被选中，提高搜索性能。

2.2 改进遗传算法在网格任务调度中的应用

在网格任务调度模型中，设 $x=\{x_1,x_2,\cdots,x_m\}$，其中 m 是任务数，x_i 是介于 $1\sim n$ 之间的一个整数，即主机编号。因此用 x 来表示一种选择方案，在遗传算法中它表示一个染色体。例如由 10 个任务和 5 台主机组成的系统中，方案[2,1,5,4,2,3,5,1,4,5]表示第 1 个任务由第 2 台主机完成，第 2 个任务由第 1 台主机完成，第 3 个任务由第 5 台主机完成，依次类推。

遗传算法通过适应度函数来确定染色体的优劣，所以必须根据实际问题的需要选择适应值函数。由于一次调度过程中，求解的任务总完成时间 *makespan* 是一个最小值，故定义适应度函数为：

$$fit(ms)=1-\frac{ms-ms_{\min}}{ms_{\max}-ms_{\min}} \tag{1}$$

其中，ms 为对应某一染色体编码的任务总完成时间，是 *makespan* 的缩写，$ms_{\min}$ 为总完成时间的最小估算值，$ms_{\max}$ 为总完成时间的最大估算值，$ms_{\min}$ 和 $ms_{\max}$ 适当取值，使 $0<fit(ms)<1$。总完成时间 ms 越小，适应度 $fit(ms)$ 越大，该染色体被选中的可能性就越大。

改进遗传算法的流程如下：

(1)随机产生满足染色体定义的初始种群，种群大小为 popsize；定义进化代数的初值和最大值，设定交叉概率、变异概率等初始参数；

(2)计算种群中每一个个体所对应的适应度函数值及负载均衡度值；

(3)将种群中负载均衡度值最低的部分个体剔除掉，并对适应度值高的个体复制，以补充被剔除掉的个体，保证总的染色体数为 popsize 不变；

(4)采用轮盘赌法进行选择操作；

(5)随机配对，按照给定的概率进行单点交叉、交叉点随机设置；

(6)对个体的每一个基因座，根据变异概率指定其变异点，对每一指定的变异点的基因值由除该基因值以外的随机产生的[1,n]之间的数值代替；

(7)寻找种群中最大适应度值和与之对应的最小完

成时间；

(8)判断算法是否达到最大迭代次数，如未满足条件，则转步骤(2)继续搜索；否则输出全局最优适应值及其对应的染色体，将该任务选择方案赋给网格调度模型。

为了验证本文提出的IGA算法的有效性，用IGA算法与基本遗传算法及经典的Min-Min算法进行对比仿真研究。

Min-Min算法[7]是网格调度算法的研究基础，该算法的目的是将大量的任务指派给完成最早、执行最快的机器。算法的思想为：对等待分配的每一个任务，分别计算出该任务分配到 n 台机器上的最早完成时间；从中找出具有最小最早完成时间的任务，并将该任务分配给对应的机器；分配完成后，更新机器期望就绪时间并删除已经分配的任务。重复上面的过程，直到分配完所有的任务。

3 仿真实验及结果分析

在Matlab环境下设计一个网格任务调度系统模拟程序，该程序可以根据仿真的需要生成不同的主机处理能力、主机数量、任务数量、每个任务的预测执行时间、通信开销及其他时间开销等参数。在用改进的遗传算法寻找任务调度的最佳方案时，种群规模为40，最大允许迭代次数为400。在本文的所有仿真实验中，针对每种问题都重复进行10次独立实验，并对实验得到的各项数据求平均值。

(1)产生一个任务数为80、主机数为8的模型，将IGA、GA和Min-Min算法同时用于该模型的任务调度，图1和图2分别表示采用IGA和GA算法时每台主机的执行时间以及完成所有任务总的执行时间情况，而图3则表示采用Min-Min算法时的仿真结果。

从仿真结果可以看出，采用IGA进行任务调度时的*makespan*是22.573 1，采用GA进行任务调度时的*makespan*是23.611 2，而采用Min-Min算法得到的调度方案的*makespan*是24.100 9，采用IGA算法比采用GA算法节省了4.40%的执行时间，采用IGA算法比采用Min-Min算法节省了6.34%的执行时间。图1中主机1、4、8的负载略高，总的负载均衡性较好；图2中主机3、8的负载低，主机2负载最高，负载均衡性不如图1；但在图3中，主机1、3、6、8负载明显高于其他主机，负载均衡性不如图1，也不如图2，这就是Min-Min算法的主要缺陷。另外，应用IGA进行调度时，负载最大的主机与负载最小的主机的负载比值为1.085 1；采用GA算法时，这个数据为1.160 6，而采用Min-Min算法时，这个比值则为1.247 9。

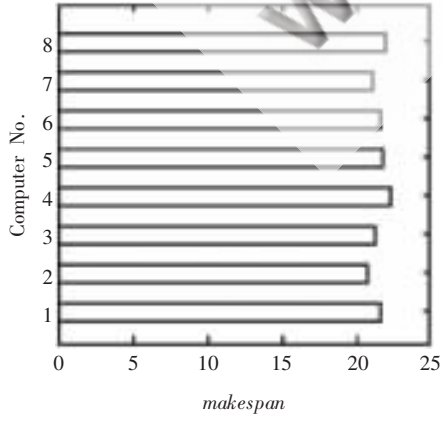

图1 IGA的调度结果

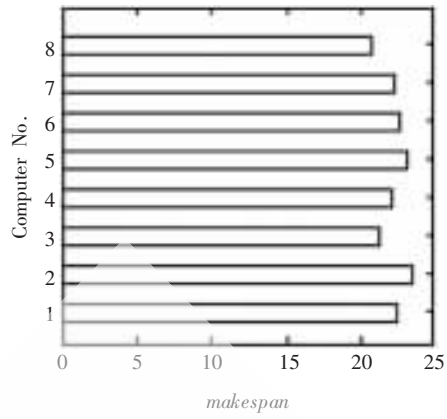

图2 GA的调度结果

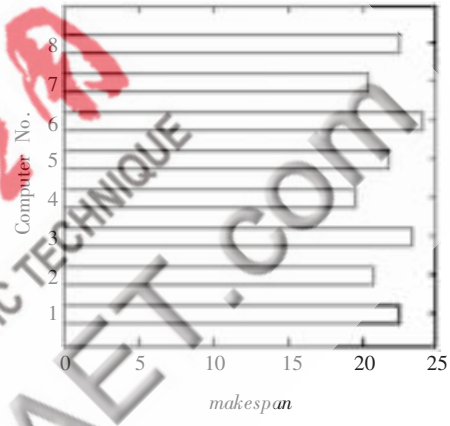

图3 Min-Min算法的调度结果

(2)在主机数量不变的情况下，改变任务数量，分别应用IGA、GA和Min-Min算法寻找最优调度方案，表1给出了两种算法对应的*makespan*的仿真结果对比，结果显示IGA优于GA和Min-Min算法。

表1 不同任务数时的完成时间

任务数	主机数	IGA	GA	Min-Min
80	8	22.573 1	23.611 2	24.100 9
100	8	30.446 8	32.092 1	33.907 1
120	8	38.984 5	40.661 0	41.885 1
140	8	44.323 2	47.009 6	48.017 7

(3)在任务数量固定的情况下，主机数量从8台均匀递增至20台，分别应用IGA、GA和Min-Min算法寻找最优调度方案，并计算三种方案的*makespan*，将仿真结果列于表2中，结果显示IGA算法比GA算法和Min-

表 2 不同主机数时的完成时间

任务数	主机数	IGA	GA	Min-Min
100	8	30.446 8	32.092 1	33.907 1
100	12	23.763 3	26.754 3	28.664 9
100	16	16.774 0	19.558 8	21.532 2
100	20	11.796 1	14.609 9	15.947 8

Min 算法得到的任务完成时间短。

在分析网格任务调度问题和基本遗传算法的基础上，提出了一种带过滤机制的改进遗传算法，并用于网格任务调度模型的优化。用 Matlab 进行仿真实验，仿真结果表明文中所提算法有很好的特性，得到了较基本遗传算法和 Min-Min 算法更为满意的结果，可以实现网格资源之间公平有效的任务调度。

参考文献

[1] BHARADWAJ V, GHOSE D, ROBERTAZZI T G. Divisible load theory: a new paradigm for load scheduling in distributed systems[J]. Cluster Comput., 2003, 6(1): 7-17.

[2] FOSTER I. The grid: blueprint for a new computing infrastructure (2nd Edition)[M]. Morgan Kaufmann Publishers Inc., ISBN: 1-55860-993-4, 2004.

[3] BRAUN T D, SIEGEL H J, 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(1): 810-837.

[4] NORIYUKI F, KENICHI H. A comparison among grid scheduling algorithms for independent coarse-grained tasks [C]. Proceedings of the International Symposium on Applications and the Internet Workshops (SAINTW'04), 2004: 674-680.

[5] CAO H Y, WANG D W. A simulation based genetic algorithm for risk-based partner selection in new product development. International Journal of Industrial Engineering, 2003, 10(1): 16-25.

[6] ÖZDAMAR L. A genetic algorithm approach to a general category project scheduling problem[J]. IEEE Trans. Syst., Man, Cybern. C., 1999, 29(2): 44-59.

[7] HE Xiao Shan, SUN X H, VON L G. QoS guided Min-Min heuristic for grid task scheduling[J]. Journal of Computer Science & Technology, 2003(5): 442-451.

(收稿日期：2010-04-27)

作者简介：

卜艳沛，女，1964 年生，副教授，主要研究方向：智能算法及其应用。