第33卷 第18期 计 算 机 工 程 2007年9月
Vol.33 No.18 Computer Engineering September 2007
·人工智能及识别技术· 文章编号：1000—3428(2007)18—0181—03 文献标识码：A 中图分类号：TP393

基于融合进化计算的网格任务调度算法

杨 博，陈志刚，刘 立
(中南大学信息科学与工程学院，长沙 410083)

摘 要：在网格计算中，任务调度是一个重要的组成部分。针对网格环境异构、分布等特点，该文结合遗传算法与蚂蚁算法的优点，在双层进化结构基础上，提出了一种基于融合进化计算的网格任务调度算法。模拟实验结果表明：在网格环境下，调度算法具有明显的优势。
关键词：融合进化计算；网格计算；任务调度；染色体

Grid Task Scheduling Algorithm Based on Combined Evolutionary Computation

YANG Bo, CHEN Zhi-gang, LIU Li
(School of Information Science and Engineering, Central South University, Changsha 410083)

【Abstract】 Task scheduling is an important part in grid computing. For heterogeneous and distributed grid, a combined evolutionary computing based grid task schedule algorithm is presented, combining the advantage of genetic algorithm with that of ant algorithm, based on a double-deck genetic structure. Simulation results show that the algorithm performs better than other scheduling algorithms obviously in grid.
【Key words】 combined evolutionary computation; grid computation; task scheduling; chromosome

在网格计算中，任务调度是一个重要的组成部分，已被证明为NP完全问题，可以用启发式方法求解。网格环境下的任务调度包括元任务的调度和依赖任务的调度。元任务独立于其他任务，而“依赖任务”存在依赖关系。当前许多网格任务调度算法是建立在元任务基础上的，考虑依赖任务的情况较少，具有一定的局限性，如最小最小算法(min-min)、最大最小算法(max-min)、最大时间跨度算法(max-int)、快速贪吃算法(fast-greedy)[1,2]等。

对于依赖任务的调度，研究者基于有向无环图(directed acyclic graph, DAG)等模型提出了一些网格调度算法，如分代算法(generation scheduling, GS)[1]、DLS(dynamic level scheduling)[3]等。在以往的研究中，产生了许多启发式调度策略，如表调度(List scheduling)、聚簇调度(clustering)、基于任务复制调度(task duplication)等，而遗传算法(genetic algorithm, GA)[4~8]和蚂蚁算法(ant algorithm, AA)[9,10]是一类智能优化方法，具有独特的优势，二者的搜索策略单独应用在任务调度算法中，各有利弊。本文将2种算法相结合，有效地提高搜索性能。

1 调度模型

依赖任务之间的关系用一个加权的DAG图来表示，公式为

$$G=(V,E,C,W)$$

其中，G代表DAG图；V,E,C,W分别代表任务节点、有向边、计算开销和通信开销的集合。

节点 $v\in V$ 表示一个计算开销为 $C(v)$的任务，有向边 $(u,v)\in E$ 表示任务的先后关系，即任务 u 完成后，任务 v 才能开始，任务 u 到任务 v 的通信开销为 $W(u,v)$。如果这2个任务分配到同一个网格资源上，则认为通信开销为0。如果存在一条从 u 到 v 的路，则称 u 为 v 的前驱节点，v 为 u 的后继节点。特别地，对于 $(u,v)\in E$，称 u 为 v 的立即前驱节点(记为 $u\in iPred(v)$)，v 为 u 的立即后继节点(记为 $v\in iSucc(u)$)，本文中任务和节点可以互换使用。

2 算法描述

2.1 融合进化任务调度思想简介

本算法基本思想是：先采用GA，根据网格任务调度的特点构造染色体、适应度函数等相关组件，充分利用GA快速性、随机性、全局收敛性进行进化求解；在进化后期收敛速度大大降低前及时停止进化过程，将所得解转换成为AA的初始信息素分布，弥补了AA初始信息素不足的缺点，大大提高了求解速度；然后，AA充分利用其并行性、正反馈性、求精解效率高等特点获得调度解，从而在优化性能和时间性能上得到提高。

2.2 染色体编码

不同于传统的单层进化结构，本算法采用了双层染色体进化编码，结构见图1。

t_1	t_2	t_3	…	t_n
r_1	r_2	r_3	…	r_n

图1 染色体结构

由图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

基金项目：国家自然科学基金资助项目(60573127)；高等学校博士学科点专项科研基金资助项目(20040533036)
作者简介：杨 博(1979－)，男，博士研究生，主研方向：网络分布式计算；陈志刚，教授、博士生导师；刘 立，博士研究生
收稿日期：2006-11-06 **E-mail：**rodmanyy2002@163.com

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，对图中各任务按高度分层排序[4]，其计算公式如下：

$$Height(v_i)=\begin{cases}0 & \text{if } i\,\mathrm{Pr}ed(v_i)=\phi\\ \max\limits_{v_j\in \mathrm{Pr}ed(v_i)}\mathrm{Height}(v_j)+1 & \text{otherwise}\end{cases} \tag{1}$$

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。

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)将每个任务按步骤(3)中得到的顺序，逐个随机映射到 m 个网格资源中的一个，生成染色体的网格资源层；

(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(4)$pSize$ 次，生成初始种群。

2.4 适应度函数

算法使用适应度函数(fitness function)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。算法函数为

$$Fitness(c_i)=\frac{1}{FT(c_i)} \tag{2}$$

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，采用轮盘赌选择方式[11]，个体c_i被选中的概率定义如下

$$p(c_i)=Fitness(c_i)\Big/\sum_{j=1}^{pSize}Fitness(c_j) \tag{3}$$

其中，$pSize$ 为种群大小。根据选择概率，适应度较高的个体容易被保留下来。

(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。

通过反复的遗传操作，实现种群的进化，调度结果向最优解收敛。

2.6 遗传迭代终止及信息素转换

本算法在GA求解效率显著降低之前，在某一恰当的时间点终止GA，将所得结果转化为AA的初始信息素分布，通过AA向最优解迅速收敛而获得最终解，该策略不但缩短了求解过程，提高了求解效率，还充分融合了二者优点，获得更好的性能[12]。

为了确保 GA 与 AA 在适当时机融合，采用了如下遗传迭代终止条件:

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

(2)设定终止代数 GAgen 和终止阈值 GAthr，在 GAloop 内，如果连续 GAgen 代，子代种群最优解的变化值都小于 GAthr，说明算法收敛显著变慢，遗传迭代终止。

GA 与 AA 融合的关键在于将 GA 求解结果转换为信息素，转换机制为：选取 GA 终止时种群中适应值最好的部分个体作为遗传优化解集合，根据集合内优化解的特征计算出可供 AA 使用的初始信息素值。信息素计算公式表示为

$$\tau_j(0)=(c\cdot\sum_{i=1}^{n}rN)/n+Cap_j+S_{para}/T_{para} \tag{4}$$

其中，rN为遗传优化解集合中每个个体内资源j被分配的任务数；n为集合大小；c为根据具体问题规模设定的参数；Cap_j为资源j的性能值；S_{para}为$para$参数长度；T_{para}为资源j到资源监控器的传输时间。

2.7 AA 计算部分

本算法 GA 部分终止后，通过 2.6 节所述方法，将所得调度解转换为信息素，进入 AA 计算部分。

蚂蚁算法通过蚂蚁留下的信息素(pheromone)强弱形成正反馈机制，使蚂蚁适应性地向最短路径集中。本算法利用该特性向具有最小完成时间的调度序列收敛。

根据 MMAS(max-min ant system)算法的信息素设置原理，结合本算法具体情况，信息素设置与更新策略规定如下：

(1)由于从 GA 结果获得了初始信息素，因此 AA 信息素的初值设置为

$$\tau_{init}=\tau_{\min}+\tau_p \tag{5}$$

其中，τ_{min}为根据任务调度规模给定的信息素常数，设定了信息素浓度的最小值；τ_p是GA求解结果转换的信息素值$\tau_j(0)$。

(2)每圈只对具有最早完成时间的蚂蚁进行信息素增加；

(3)将各调度方案的信息素浓度限制在[$\tau_{\min}$,$\tau_{\max}$]，超出这个范围的值则相应设为$\tau_{\min}$或者$\tau_{\max}$，以防止算法过早收敛于非全局最优解。

信息素τ_j的更新方程为

$$\tau_j^{new}=\begin{cases}(1-\rho)\cdot\tau_j^{old}+\Delta\tau_j^{best} & \text{本圈最佳调度中任务从资源}j\text{成功返回，且}\tau_{\min}<\tau_j^{new}<\tau_{\max}\\ \tau_{\max}, & \tau_j^{new}>\tau_{\max}\\ \tau_{\min}, & \tau_j^{new}<\tau_{\min}\\ (1-\rho)\cdot\tau_j^{old}, & \text{其他}\end{cases} \tag{6}$$

其中，ρ 为信息素挥发率；$\Delta\tau_j^{best}$为本圈中最佳蚂蚁(即最快完成调度的蚂蚁)在该调度序列上信息素的增加量。

在 t 时刻，任务被分配到资源 j 的概率表示为

$$p_j(t)=\frac{[\tau_j(t)]^{\alpha}\cdot[\eta_j]^{\beta}}{\sum_k[\tau_k(t)]^{\alpha}\cdot[\eta_k]^{\beta}} \tag{7}$$

其中，$\tau_j(t)$是在t时刻从调度中心到资源j路径上的信息素；η_j是资源j的先天的性能，$\tau_j(0)$；$\tau_k(t)$是在t时刻从调度中心到资源k路径上的信息素；η_k是资源k的先天的性能，即$\tau_k(0)$；α是信息素的控制因子；β是资源先天属性的控制因子[10]。

2.8 CEGTS 算法总体描述

(1)初始化算法控制参数；

(2)设置 GA 及 AA 计算结束条件；

(3)生成初始种群并计算初始种群中个体的适应值；

(4)反复执行选择操作、交叉操作、变异操作，直到满足GA结束条件；

(5)从GA结果中选择适应能力强的前15%个体放入集合中，作为优化解集合；

(6)将优化解集合转换为AA的初始信息素；

(7)将 m 只蚂蚁分布在各个资源上(m 取任务数)，每只蚂蚁根据当前各资源的信息素值决定自己解域的下一任务分配给剩余各资源的概率，确定可行的调度方案；

(8)计算 m 种调度序列的适应值，并从中选择最佳方案；

(9)根据式(6)，更新所有资源的信息素值；

(10)若满足AA结束条件，则综合GA优化解结果报告最佳调度方案并退出；否则，继续执行(7)。

3 实验

在相同的模拟环境下，分别用本算法与基于遗传算法GA的网格任务调度算法、基于蚂蚁算法AA的网格任务调度算法及分代算法GS[1]进行计算比较。算法主要参数为：种群大小为150，算法的杂交概率0.75，变异概率为0.05。实验结果见图2和图3。

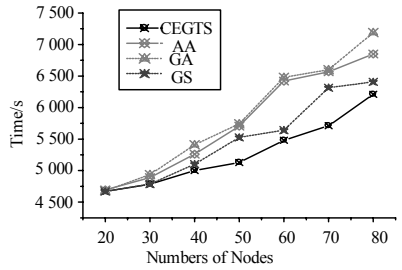

图2 CEGTS、AA、GA、GS运算时间比较

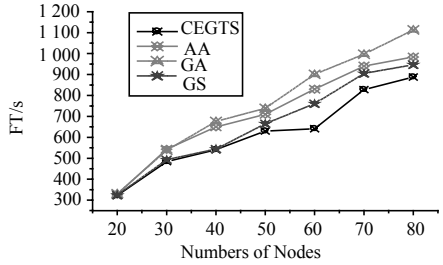

图3 CEGTS、AA、GA、GS求解结果比较

图2表示的是CEGTS,GA,AA,GS 4种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，任务节点较少时，4种算法运算时间相差不多，随着任务数增多，本算法运算时间则明显少于其他3种算法，即求解速度更快。图3表示的是该4种算法在不同任务数情况下的求解结果比较图，可以看出，本算法得到的调度完成时间小于其他3种算法，而且随着任务数增加，本算法的求解结果越优于其他3种算法。

4 结论

本文提出了一种基于融合进化计算的网格任务调度算法。本算法充分利用了遗传算法和蚂蚁算法的优点，将融合进化计算用于网格任务调度，提高了调度性能，缩短了运算时间。实验结果表明，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算法。

参考文献

1 桂小林. 网格技术导论[M]. 北京: 北京邮电大学出版社, 2005.

2 Braun T D. A Comparison Study of Static Mapping Heuristics for a Class of Meta-Tasks on Heterogeneous Computing Systems[C]//Proc. of IPPS/SPDP Workshop on Heterogeneous Computing, San Juan, Puerto Rico. 1999-04.

3 Sih G C, Lee E A. A Compile-time Scheduling Heuristic for Interconnection-constrained Heterogeneous Processor Architectures[J]. IEEE Transactions on Parallel and Distributed Systems, 1993, 4(2): 308-323.

4 钟求喜, 谢 涛, 陈火旺. 基于遗传算法的任务分配与调度[J]. 计算机研究与发展, 2000, 37(10): 1197-1203.

5 Yao Wensheng. Genetic Scheduling on Minimal Processing Elements[M]. Berlin Heidelberg: Springer-Verlag, 2002.

6 Lee Y H, Chen C. A Modified Genetic Algorithm for Task Scheduling in Multiprocessor Systems[C]//Proceedings of the 9th Workshop on Compiler Techniques for High-performance Computing. 2003.

7 Wu A S, Yu H, Shiyuan J, et al. An Incremental Genetic Algorithm Approach to Multiprocessor Scheduling[J]. IEEE Transactions on Parallel and Distributed Systems, 2004, 15(9): 824-834.

8 Martino V D. Scheduling in a Grid Computing Environment Using Genetic Algorithms[C]//Proc. of the 16th Int'l Parallel and Distributed Processing Symp. Conference, Florida. 2002.

9 Ritchie G, Levine J. A Hybrid Ant Algorithm for Scheduling Independent Jobs in Heterogeneous[C]//Proc. of the 23rd Workshop of the UK Planning and Scheduling Special Interest Group. 2004.

10 Xu Zhihong, Hou Xiangdan, Sun Jizhou. Ant Algorithm-based Task Scheduling in Grid Computing[C]//Proc. of Electrical and Computer Engineering Conference. 2003.

11 钟一文, 杨建刚. 异构计算系统中独立任务调度的混合遗传算法[J]. 北京航空航天大学学报, 2004, 30(11): 1080-1083.

12 熊志辉, 李思昆, 陈吉华. 遗传算法与蚂蚁算法动态融合的软硬件划分[J]. 软件学报, 2005, 16(4): 503-512.

(上接第180页)

6 Anupriya A. DAML-S: Web Service Description for the Semantic Web[C]//Proceedings of the 1st International Semantic Web Conference. 2002.

7 Klein M, Bernstein A. Searching Services on the Semantic Web Using Process Ontologies[C]//Proceedings of the International Semantic Web Working Symposium. 2001: 159-172.

8 Trastour D, Bertolini C, Javier G C. A Semantic Web Approach to Service Description for Matachmaking of Services[C]//Proc. of the International Semantic Web Working Symposium. 2001: 447-461.