基于并行遗传蚁群混合算法的网格资源调度研究①

Research on Resource Scheduling Based on Parallel Genetic Ant Colony Hybrid Algorithm in Computational Grid

朱　英[1]　雷领红[1,2]　黄文明[1]

（1.桂林电子科技大学 计算机与控制学院;2.桂林电子科技大学 信息科技学院 广西 桂林 541004）

摘　要: 合理的资源调度算法可以在很大程度上提高网格系统的有效利用率。在网格环境中,用户希望自己的任务尽可能快地得到完成,而网格环境则希望尽可能充分地利用所有资源。针对这种情况,本文提出了基于并行遗传蚁群混合算法的网格资源调度,具有较小的时间复杂度和良好的调整性能。实验证明了其正确性与实用性。

关键词: 网格 资源调度 混合算法

关于网格资源调度,国内外已做了许多研究工作,先后提出了各种调度算法。这些算法按照调度策略可以分为动态调度和静态调度两种[1]。静态调度算法需要花费大量时间计算任务调度表,算法缺少灵活性,如任务添加、删除或任务特征变化,都需要重新计算调度表,而且静态调度算法每隔一定周期进行一次调度,因此越早到达的任务等待的时间越长,从而使得任务的响应时间过长。相比之下,动态调度算法的环境适应性好,在多种环境下操作性能良好,算法灵活。因此,研究高性能的适应网格环境的动态资源调度算法具有一定的现实意义。

网格环境下的资源调度是提高网格性能的关键技术之一,资源调度的目的是在包含大量不同计算机的网格环境中,把不同的任务以最合理的方式分配给相应的网格节点去完成。由于网格中的资源是动态变化的,可以随时加入或退出,因此网格资源调度算法应具有分布性,可扩展性和容错性[2]。算法效率的好坏直接影响到网格环境的整体性能,也关系到网格服务质量的好坏。

遗传算法是一类基于自然选择与遗传学原理的有效搜索方法,它从一个种群开始,利用选择、交叉、变异等遗传算子对种群进行不断进化,最后得到全局最优解[3]。而蚂蚁算法具有较强的健壮性和内在的分布并行性,且易于和其他方法相结合,适合网格资源调度要求[4]。蚂蚁算法还有一个优点就是具有可扩展性。文中将两种算法有效的混合生成新算法,使用遗传算法具有快速随机的全局搜索能力生成信息素分布,再利用蚁群算法求最优解,使之更适应网格资源调度。

1　混合算法设计思想

在问题描述之初,我们对资源进行初始化编码,选择,交叉,变异操作之后,把资源的初始信息素量赋给当前网格中的资源。然后采用蚁群算法来搜索当前分配策略中的最佳策略。算法的基本过程和求解 TSP 问题的蚁群算法相似,但也有不同之处,主要问题在以下三个方面：

(1) 基于 TSP 的蚁群算法中城市之间有边相连,而且这些边是可达的并且有不同的距离,而资源网格分配中没有这样的拓扑关系。所以在网格环境中要利用资源活动的相互作用来模拟拓扑关系,以供蚁群算法使用。

(2) 在求解 TSP 问题是,城市与城市之间的信息素用城市之间的距离来表示,而在资源网格分配中用资源的计算和通讯能力等相关系数来表示信息素。

① 基金项目 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

(3)蚁群算法中,启发信息也起着很重要的作用。我们也需要以资源固有的属性(计算和通讯能力)来表示在蚁群算法中的启发信息。

2 混合算法设计过程

2.1 混合算法动态临界点的确定

如何寻找两种算法的最佳结合点是混合算法的关键所在。下面的动态融合策略可以确保遗传算法与蚁群算法在最佳时机融合。

(1)设置最小遗传迭代次数(如 t_b 时刻)和最大遗传迭代次数(如 t_c 时刻)。

(2)遗传算法迭代过程中统计子代群体的进化率,并以此设置子代群体最小进化率。

(3)在设定的迭代次数范围内,如果连续 N 代,子代群体的进化率都小于最小进化率,说明这时遗传算法优化速度较低,因此可终止遗传算法过程,进入蚁群算法。

2.2 遗传算法部分

(1)编码

本文针对网格资源分配的特点,采用树型编码方式。在本算法中,每个个体(染色体)是覆盖源节点和目的节点集合的子树的叶子节点,即个体采用树型结构,遗传操作直接作用在这样的个体上,其优点在于树型编码使得遗传操作意义直观,并且不用进行编码空间和解空间的转换,节省算法时间。

(2)群体初始化

遗传算法中个体的生成采用随机深度优先搜索算法,其基本做法是:从源节点开始,随机地选择一个与之相关联的资源节点,将两点相连;然后从选择的节点继续随机地选择与其关联的下一个资源节点,在连接时要判断条件是否满足,如果不满足则继续选择节点,这样做下去直至搜索到所有满足条件的目的节点。

(3)适应度函数

适应度函数是遗传算法用来判断个体好坏的标准,本算法中个体是一棵覆盖源节点和目的节点的子树,如果所用时间越少,表明节点的性能越好,其适应度应该越高,同时认为时延小的个体的适应度值要高,为此,本算法设定节点 X 的适应度函数为 $f(x)=\frac{1}{\sum_{e\in T} c(e)*\delta}$,这里 $c(e)$ 表示子树中边 e 的所耗时间,$\delta=\max_{j\in k}|\sigma-R_i|$,其中 σ 是目的节点的时延约束值,R_i 表示节点集合中从资源节点 i 到相连节点 j 的时间。从适应度函数来看,节点的耗时越少,适应度值就会越大。

(4)选择操作

选择算子就是为了将父代的个体信息传递到子代。每代中的节点都按照适应度函数的大小决定它能够复制到下一代的概率。通过选择,使得群体中的优秀个体数目不断增加,整个进化过程朝着更优解的方向进行,体现了自然选择中的进化原则。我们采取以下公式选择个体,第 i 个个体 X_i 被选择的概率为:

$$P_n=\frac{f(X_i)}{\sum f(X_i)}$$

其中 $f(X_i)$ 为适应度函数。

(5)交叉操作

交叉操作是从当代群体中随机地选择两个个体,依照下述的交叉规则以概率 1 交叉产生一个新孩子个体。该过程一直重复到下一代个体数与初始群体数目相同为止。交叉规则如下:①选择父代中相同的链路,将其直接遗传到孩子。下一步就是连接这些子树成为一棵完整的树。②连接这些零散子树前首先对这些零散子树进行分类,第一类子树包含源节点,第二类子树包含目的节点,第三类子树即不包含源节点,也不包含目的节点。

根据前面所述(优先规则),混合算法用于求从网格当前资源中选取下一资源,直至作业完成。选取资源的目标是使网格系统的运行效率高,作业吞吐量最大,系统的效率最高。

(6)变异操作

交叉完成后,新的子女个体以概率 P_m 进行变异,变异概率取 0.05。变异规则如下:从新孩子个体中,随机选择一些中间节点(非组播源节点和目的节点),删除连接这些节点的路径形成零散子树,然后依照与交叉操作相类似的连接方法连接各零散子树。

2.3 蚁群算法部分

信息素的初始值是把各路径信息素的初值设置为最大值,这里通过遗传算法得到的一定的路径的信息素值,我们以此来模拟蚂蚁的分泌物。蚂蚁在行进的过程中,根据各条路径上的信息素强度来决定下一步

所行进的路径，采用$p_{jk}^{i}(t)$表示在时刻 t 蚂蚁 i 由 j 节点转到节点 k 的概率，则

$$p_{jk}^{i}(t)=\begin{cases}\dfrac{\tau_j^{\alpha}(t)\eta_j^{\beta}}{\sum_k \tau_k^{\alpha}(t)\eta_k^{\beta}} & j,k\in grid\ resource\\ 0 & others\end{cases} \quad (1)$$

随着时间的推移，保留下来的信息素逐渐消逝，用参数来表示，所以搜寻一次后的信息素根据更新方程进行调整：

$$\tau_{jk}(t+1)=\rho*\tau_{jk}(t)+\sum_{i=1}^{m}\Delta\ t_{jk} \qquad \rho\in(0,1) \quad (2)$$

其中$\tau_{jk}(t)$为当前节点的信息素，$\tau_{jk}(t+1)$是下一时刻的节点信息素，随着时间的推移，信息素会变化，节点的路由表信息也会随之更新。

2.4 目标函数及终止条件

混合算法终止的条件：

$$\min\sum_{i=1}^{N} f_i(x_i) \quad (3)$$

$$s.t.\ \sum g_i(x_i)\le Rg\ \text{其中}\ {}_i(x_i)\le r_i, i=1,2...,N$$

公式(3)中，R 为当前网格中的资源，r_i 为各子网格域系统所占资源。当所求的节点集合满足以上条件时，算法结束。

3 混合算法工作流程与操作过程

3.1 算法工作流程图

算法工作流程图如图 1 所示。

3.2 算法操作过程

下面伪代码说明了混合算法的操作过程：

begin

初始化，选择问题的编码方式

初始化，确定问题各参数的值

for each h in 种群中每个个体

计算每个个体适应度

end of for

Whil 不满足结束条件时 do

选择操作

交叉操作

变异操作

for each h in 种群中每个个体

计算每个个体适应度

end of for

end of while

评价当前最优的 N 个个体

将 N 个个体转化为蚂蚁出行的信息素浓度矩阵

While 不满足结束条件时 do

for each s in 蚁群中的每个蚂蚁

for each t in 每个解构造步(直到构造出完整解)

初始化蚂蚁已访问节点列表

初始化蚂蚁未访问节点列表

确定蚂蚁即将访问节点列表

计算即将访问节点的信息素浓度，确定下一个访问节点

回到出发点，更新每只蚂蚁所走过路径的信息素浓度

end of for

end of for

end of while　比较所得的可行解　输出最优解

End

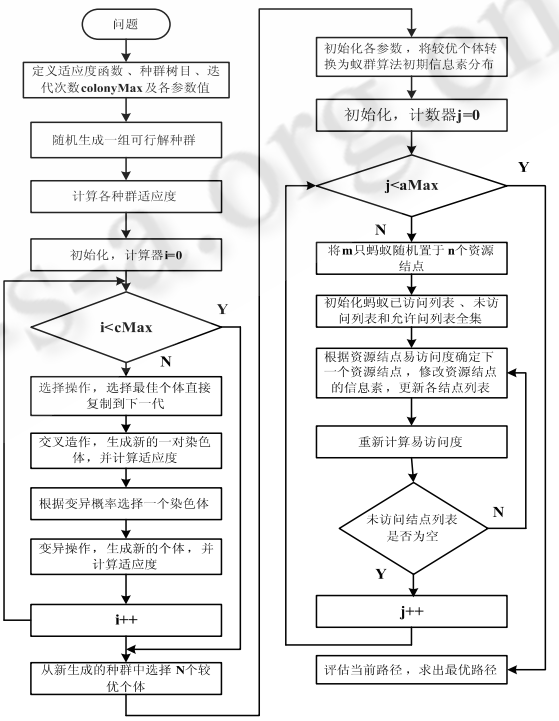

图 1　算法工作流程图

4 试验结果与分析

为了证明资源代理程序和相关调度算法的有效

性,需要在不同情况下评估它们的性能,如改变资源数和不同要求的用户。本文利用 GridSim 工具采用实验分析算法,对混合算法与基本蚁群算法进行对比分析。文献 5 中所使用的随机网络拓扑结构生成法所产生的网络结构可以具有实际网络的一些特征。在仿真实验中所采用的网络结构由这种方法生成,设定资源节点数为 8,算法中遗传算法的迭代次数为 25 次,蚁群算法中各路径信息素初始值为 τ_0 设为 50。混合算法求解过程中 α、β、ρ 值的设定根据表 1。为测试混合算法的性能,我们分别利用混合算法和基本蚁群算法进行优化,所得到的实验结果如表 1 与图 2 所示,该结果进一步表明了混合算法在有效性方面的优越性能。

表 1　基本蚁群算法和混合算法实验结果

α	β	ρ	算法的进化代数	
			基本蚁群算法	混合算法
1	1	0.8	355	25 +13
1	2	0.8	345	25 +11
2	1	0.8	386	25 +18
2	2	0.8	370	25 +15
2	3	0.8	394	25 +21
3	2	0.8	387	25 +19
3	3	0.8	342	25 +10

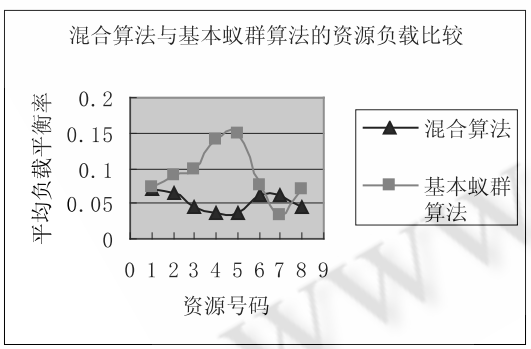

图 2　混合算法与蚁群算法的资源负载比较

5　结语

混合算法是将并行遗传算法快速的全局搜索能力与蚁群算法高效的局部最优性质相结合生成的,通过实验说明混合算法具有比基本蚁群算法更强的搜索全局最优解的能力。而且混合算法比基本蚁群算法有更好的资源负载平衡率、稳定性和收敛性。

参考文献

1 Fujimoto N, Hagihara K. A comparison among grid scheduling algorithms for independent coarse – grained tasks. In: Proc. Of the 2004 Symp. on Applications and the Internet – Workshops. Washington: IEEE Computer Society Press, 2004. 674 – 680.

2 De Turck F, Vanhaste S. A generic middleware – based platform for scalable cluster computing J, Future Generation Computer Systems, 2002, 1: 549 – 560.

3 Srinivas M, Patnaik L M. Genetic algorithm: a survey. IEEE Computer, 1994, 27(6): 17 – 26.

4 Liang Y C, Smith A C. An ant system approach to redundancy allocation C, Proceedings of the Congress on Evolutionary Computation, Washington, 1999, 1478 – 1484.

5 段海滨. 蚁群算法原理及其应用. 北京: 科学出版社, 2005: 35 – 36.

(上接第 91 页)

Inter Conf on Robotics and Automation. 1988(3): 1315 – 1317.

13 Lee S S, Williams J H. A fast tracking error control method for an autonomous mobile robot. Robotica. 1993, 11: 209 – 215. Reference 5.

14 Waxman A M, Lwmoigne J J. A visual navigation system for autonomous land vehicles. IEEE Journal of Robotics and Automation, 1987, RA – 3: 349 – 358.

15 Tomas W. Miller, III. Real – Time Application of Neural Networks for Sensor – Based Control of Robots with Vision. IEEE Transactions on System, Man and Cybernetics, 1989, 19(4): 825 – 831.