第16卷 第3期
2006年3月

计算机技术与发展
COMPUTER TECHNOLOGY AND DEVELOPMENT

Vol.16 No.3
Mar. 2006

基于蚂蚁算法的网格任务分配算法研究

陈 歆,罗四维
(北京交通大学 计算机与信息技术学院,北京 100044)

摘 要:任务分配是网格计算环境中影响系统性能的重要因素,由于网格环境中各种资源分布于不同的地理位置,其性能也千差万别,因此需要一种有效的策略来进行任务的分配,使得整个系统完成任务的代价最小。文中将蚂蚁算法应用于解决网格环境中的任务分配问题,并进行了仿真实验,取得了良好的效果。

关键词:网格;任务分配;蚂蚁算法
中图分类号:TP301.6 **文献标识码**:A **文章编号**:1005-3751(2006)03-0098-03

Research of Grid Computing Task Assignment Algorithm Based on Ant Algorithm

CHEN Xin, LUO Si-wei
(School of Computer and Information Technology, Beijing Jiaotong University, Beijing 100044, China)

Abstract: Task assignment is a key factor that will affect the performance of computing grid. All kinds of resources are located all over the world, and the performances differ in thousand ways, so it's necessary to find an effective task assignment policy to minimize the cost of the whole grid. This thesis applies the ant algorithm to solve the task assignment problem in grid, and implements related experiments, got fair result.

Key words: grid; task assignment; ant algorithm

0 引 言

网格被称为下一代的互联网,它能够聚合各种计算资源,并将其转化为一种随处可得的、可靠的、标准的同时还是经济的计算能力[1]。但是由于网格中的计算资源在地域上呈现出分散分布的特点,而且各个计算资源之间也不尽相同,具有异构性、动态性和不稳定性,因此任务的分配和调度就成为网格环境中十分重要的一部分,它的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。

1 网格环境中的任务分配问题

网格的多机环境下的任务分配问题是一个典型的组合优化问题,同时也是一个NP问题,今年来出现的一些启发式算法(如模拟退火、遗传算法以及混沌神经网络等)为解决此类问题提供了方法。

蚂蚁算法(ant algorithm)[2]是意大利学者Dorigo M等人首先提出来的,它是一种源于生物世界的一种仿生类方法,作为随机型的优化方法,它吸收了蚂蚁的行为特点,通过其内在的搜索机制,在许多困难的优化组合问题的求解中都取得了很好的成效。据昆虫学家观察和研究发现,蚂蚁能在没有任何外界提示的情况下,找到从蚁巢出发到达食物源的最短路径,并能随着外部环境的变化,搜寻新的路径,做出新的选择。蚂蚁在寻找食物源的过程中,会在自己走过的路线上留下蚂蚁特有的分泌物——信息激素(pheromone),使得其他蚂蚁能够察觉,并由此影响它们以后在搜索过程中的行为。当一些路径上的蚂蚁越来越多时,它们留下的信息激素也就越来越多,浓度也越来越大,期间信息激素也会随着时间不断挥发,蚂蚁选择这条路径的概率也就不断增大,从而增加了这条路径的信息激素的强度,由此可见,其原理是一种正反馈机制。

蚂蚁算法是一种解决NP问题的有效方法,具有较强鲁棒性和内在的分布并行性。此外蚂蚁算法还具有一个很大的优点——可扩展性。可扩展性就是指在原有的规模为 n 的问题上求出最优解后,再增加 m 个节点,可在原有解的基础上很快找到该问题的 $m+n$ 规模的最优解[3],因此蚂蚁算法能很好地适应网格环境中动态、不稳定的特点。

2 任务分配问题的数学模型

一个典型的任务分配问题可以描述为:有 n 项任务需要分配到 n 个节点上去完成,每个节点只能处理一个任

收稿日期:2005-06-04
作者简介:陈 歆(1981—),男,重庆人,硕士研究生,研究方向为网格计算;罗四维,教授,博士,博士生导师,研究方向为神经网络、多媒体、计算机体系结构。

务,且一个任务只能由一个节点来完成。不同的分配方案花费的代价不同,任务分配问题就是要找到一种分配方案,使得花费的总体代价最小。

设第 i 个节点完成第 j 项任务的代价为 $C_{ij}\geqslant 0$,则可构成代价矩阵 $\boldsymbol{C}_{n\times n}$。求应该如何分配,才能使完成所有任务的总体代价最小。

设

$$R_{ij}=\begin{cases}1 & \text{表示由第 } i \text{ 个节点来完成第 } j \text{ 项任务}\\ 0 & \text{表示不由第 } i \text{ 个节点来完成第 } j \text{ 项任务}\end{cases} \tag{1}$$

$i,j=1,2,\cdots,n$

则任务分配问题是求解任务分配矩阵 $\boldsymbol{R}_{n\times n}$,约束条件为:

$$\sum_{j=1}^{n}R_{ij}=1 \quad i=1,2,\cdots,n \tag{2}$$

$$\sum_{i=1}^{n}R_{ij}=1 \quad j=1,2,\cdots,n \tag{3}$$

$R_{ij}=0$ 或 1　$i,j=1,2,\cdots,n$

3　用蚂蚁算法求解任务分配问题

任务分配问题是在给出两列元素以及任意两列元素配对时的代价,找到其中一组一一对应的配对方案,使其代价和最小。设 $i=1,2,\cdots,n$ 表示节点编号,$j=1,2,\cdots,n$ 表示任务编号,一一对应的分配指每个节点分配到一项任务,每项任务也被分配到一个节点上去处理;$C_{ij}\in \boldsymbol{C}_{n\times n}$,表示节点 i 处理任务 j 的代价;设有 n 只蚂蚁,用蚂蚁的一次旅行表示一次任务分配过程,在蚂蚁的一次旅行中,蚂蚁需要走 n 步,每走一步表示分配一项任务,蚂蚁所走的步数记为 s;当所有蚂蚁完成一次旅行,视为算法循环一次;用 N_C 表示算法循环的次数;为了把蚂蚁算法应用于求解任务分配问题,引入 3 个二维矩阵 $\boldsymbol{T}_{n\times n}$,$\boldsymbol{V}_{n\times n}$和$\boldsymbol{R}^k_{n\times n}$,$k$ 表示第 k 只蚂蚁;元素 $T_{ij}\in \boldsymbol{T}_{n\times n}$,表示任务 j 被分配给节点 i 的信息激素值,初始值为常数 t;元素 $V_{ij}\in \boldsymbol{V}_{n\times n}$,表示把任务 j 分配给节点 i 的效率值,初始值为 $1/C_{ij}$;元素 $R^k_{ij}=1(R^k_{ij}\in \boldsymbol{R}^k_{n\times n})$表示在第 k 只蚂蚁的任务分配过程中,用第 i 个节点来处理第 j 项任务;矩阵 $\boldsymbol{R}^k_{n\times n}$表示第 k 只蚂蚁完成任务分配后得到的任务分配方案,其初始值为 0 矩阵;引入两个禁忌集合 $Task=\{task_1,task_2,\cdots,task_n\}$和 $Node=\{node_1,node_2,\cdots,node_n\}$,元素 $task_j\in Task$,表示第 j 项任务有待分配,同理元素 $node_i\in Node$ 表示第 i 个节点尚未分配到任务;引入一个 n 维向量 $\boldsymbol{D}^{N_C}_n$,元素 $D^{N_C}_k\in \boldsymbol{D}^{N_C}_n$,表示在第 N_C 次算法循环中,第 k 只蚂蚁进行任务分配所得到的代价向量,初始值为 0 向量;最后引入至关重要的概率矩阵 $\boldsymbol{P}^k_{n\times m}$,元素 $P^k_{ij}\in \boldsymbol{P}^k_{n\times n}$,表示在第 k 只蚂蚁分配任务的过程中把第 j 项任务分配给第 i 个节点的概率,P^k_{ij}与先前蚂蚁把第 j 项任务分配给第 i 个节点的信息激素和第 i 个节点处理第 j 项任务的效率有关,P^k_{ij}可由式(4)计算得出:

$$P^k_{ij}=\frac{T_{ij}^{\alpha}\times V_{ij}^{\beta}}{\sum_{j=1}^{n}T_{ij}^{\alpha}\times V_{ij}^{\beta}} \tag{4}$$

其中:α,β 为可调系数,α 表示与先前蚂蚁把第 j 项任务分配给第 i 个节点的信息激素的相关性;β 表示与第 i 个节点处理第 j 项任务的效率的相关性。

算法描述如下:

初始化 $N_C=1$

Repeat

For (k=1; k<=n; k++) //让所有蚂蚁都完成一次任务分配过程

Begin

禁忌集合 *Node*, *Task* 初始化;

For(s=1; s<=n; s++)//每只蚂蚁走 n 步,表示分配 n 项任务

Begin

第 1 步 从禁忌集合 *Node* 随机选出元素 $node_i$,计算 $node_i$ 处理所有任务 $task_j\in Task$ 的概率P^k_{ij},从中选出最大的,记为 $P^k_{ij_{Max}}$,置 $R^k_{ij_{Max}}=1$

从禁忌集合 *Node* 中删除 $node_i$,从禁忌集合 *Task* 中删除 $task_j$

代价向量 $\boldsymbol{D}^{N_C}_n$ 的元素 $D^{N_C}_k=D^{N_C}_k+C_{ij}$

End

End

For (k=1; k<=n; k++) //每只蚂蚁更新它走过路径上的信息素

Begin

第 2 步 使用局部更新规则。引入蚂蚁的信息素增量 ΔT,蚂蚁的信息素与其所得到的代价成反比,因此 ΔT 按照式(5)来计算:

$$\Delta T=\sum_{k=1}^{n}\frac{Q}{D^{N_C}_k} \tag{5}$$

根据第 k 只蚂蚁产生的$\boldsymbol{R}^k_{n\times n}$矩阵,如果 $R^k_{ij}=1$,则置 $T_{ij}=T_{ij}+\Delta T$

End

第 3 步 使用全局更新规则。更新矩阵 $\boldsymbol{T}$ 的值,由于信息激素的值不是不断增加的,为了防止信息激素无限增长,设置一个蒸发系数 $0<\rho<1$,以此来限制信息激素的无限增长,$\boldsymbol{T}$ 可由式(6)计算得到:

$$\boldsymbol{T}=\boldsymbol{T}\times(1-\rho) \tag{6}$$

第 4 步 求出代价向量 $\boldsymbol{D}^{N_C}_n$ 中最小的元素 $D^{N_C}_{Min}<D^{N_C}_{Min^1}$,如果 $D^{N_C}_{Min}<D^{N_C-1}_{Min}$,则置 $D_{Min}=D^{N_C}_{Min}$,$N_C=N_C+1$

Until $N_C\geqslant N_{CMax}$

4　仿真实验

实验 1[4]:将 7 个任务分配给 7 个节点($n=7$),其代价矩阵为 $\boldsymbol{C}_{7\times 7}$,已知总的代价的最优解为 167。仿真时采用参数为:$N_{CMax}=500$,$Q=1$,$\alpha=1$,$\beta=5$,$\rho=0.2$,仿

真后得到的任务分配矩阵为 $\boldsymbol{R}_{7\times7}$，求得 $D_{Min}=167$，实验结果见表 1。

$$\boldsymbol{C}_{7\times7}=\begin{bmatrix}68&68&93&38&53&83&4\\6&53&67&1&38&7&42\\68&59&93&84&53&10&65\\42&70&91&76&26&5&73\\33&65&75&99&37&25&98\\72&75&65&8&63&88&27\\44&76&48&24&28&36&17\end{bmatrix}$$

$$\boldsymbol{R}_{7\times7}=\begin{bmatrix}0&0&0&0&0&0&1\\1&0&0&0&0&0&0\\0&1&0&0&0&0&0\\0&0&0&0&0&1&0\\0&0&0&0&1&0&0\\0&0&0&1&0&0&0\\0&0&1&0&0&0&0\end{bmatrix}$$

表 1 实验结果

实验次数	D_{Min}	N_C
1	167	7
2	167	5
3	167	3
4	167	2
5	167	2
		平均值 3.8

实验 2[5]：将 10 个任务分配给 10 个节点（$n=10$），其代价矩阵为 $\boldsymbol{C}_{10\times10}$，已知总的代价的最优解为 29。仿真时采用参数为：$N_{CMax}=500$，$Q=1$，$\alpha=1$，$\beta=5$，$\rho=0.2$，仿真后得到的任务分配矩阵为 $\boldsymbol{R}_{10\times10}$，求得 $D_{Min}=29$，实验结果见表 2。

$$\boldsymbol{C}_{10\times10}=\begin{bmatrix}3&3&10&9&5&2&11&2&11&5\\6&2&7&11&4&10&4&4&5&4\\9&7&9&10&4&4&5&5&4&5\\8&6&7&8&8&8&10&6&3&9\\7&2&8&6&10&9&6&6&11&10\\5&11&3&6&10&3&6&7&2&10\\4&11&11&5&9&11&7&9&10&11\\11&10&5&4&11&4&7&8&7&3\\11&5&5&3&2&5&7&10&7&3\\10&4&5&2&11&6&11&7&8&2\end{bmatrix}$$

$$\boldsymbol{R}_{10\times10}=\begin{bmatrix}0&0&0&0&0&0&0&1&0&0\\0&0&0&0&0&0&1&0&0&0\\0&0&0&0&0&1&0&0&0&\\0&0&0&0&0&0&0&0&1&0\\0&1&0&0&0&0&0&0&0&0\\0&0&1&0&0&0&0&0&0&0\\1&0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0&1\\0&0&0&0&4&0&0&0&0&0\\0&0&0&1&0&0&0&0&0&0\end{bmatrix}$$

表 2 实验结果

实验次数	D_{Min}	N_C
1	29	10
2	29	5
3	29	9
4	29	2
5	29	6
		平均值 3.8

根据仿真实验结果可以看出：

(1) 算法简练、易懂；

(2) 具有并行算法的特点，易于扩展，适合网格环境动态变化的特点；

(3) 收敛速度快，能在较短的时间内求得最优解。

5 结 论

文中利用蚂蚁算法在求解 NP－Complete 问题方面的优势，以及其可扩展的特点，将其应用于解决异构、动态的网格环境中的任务分配问题，通过仿真实验的验证，取得了良好的效果。

参考文献：

[1] 都志辉，陈 渝，刘 鹏．网格计算[M]．北京：清华大学出版社，2002.

[2] Dorigo M, Gambardells L M. Ant Colonied for the Travelling Salesman Problem[J]. Biosystems, 1997, 43(2): 73－81.

[3] 侯向丹．蚂蚁算法扩展性及应用研究[J]．河北工业大学学报，2002(3)：15－16.

[4] Wang Xiuhong, Wang Zheng'ou, Qiao Qingli. Solving assignment problems with chaotic neural network[J]. Journal of Systems Engineering, 2001, 16(2): 146－150.

[5] Foulds L R. Combinatorial Optimization for Undergraduates [M]. Shen Minggang Translated. Shanghai: Shanghai Translation and Publishing Corporation, 1988.

(上接第 97 页)

168－196，354－373.

[2] 汤子瀛．计算机操作系统[M]．西安：西安电子科技大学出版社，2001.26－101.

[3] 毛德操，胡希明．Linux 内核源代码情景分析[M]．杭州：浙江大学出版社，2001.263－414.

[4] linux 内核源代码，v2.4.22[EB/OL]. www.kernel.org, 2003－08－25.

[5] linux 内核源代码，v2.6.10[EB/OL]. www.kernel.org, 2004－12－24.

[6] 赵 炯．linux 内核完全注释[M]．北京：机械工业出版社，2004.36－129.