第 34 卷 第 6 期 计 算 机 工 程 2008 年 3 月
Vol.34 No.6 Computer Engineering March 2008
·人工智能及识别技术· 文章编号：1000—3428(2008)06—0214—02 文献标识码：A 中图分类号：TP18

求解独立任务调度的离散粒子群优化算法

陈 晶，潘全科
(聊城大学计算机学院，聊城 252059)

摘 要：针对独立任务调度问题，提出一种改进的离散粒子群算法，采用基于任务的编码方式，对粒子的位置和速度更新方法进行重新定义。为防止粒子群算法的早熟收敛，给出利用模拟退火算法的局部搜索能力在最优解附近进行精细搜索，以改善解的质量。仿真结果表明，与遗传算法和基本粒子群算法相比，该混合算法具有较好的优化性能。

关键词：独立任务调度；粒子群算法；模拟退火算法

Discrete Particle Swarm Optimization Algorithm for Solving Independent Task Scheduling

CHEN Jing, PAN Quan-ke
(School of Computer Science, Liaocheng University, Liaocheng 252059)

【Abstract】 An improved discrete Particle Swarm Optimization(PSO) algorithm is presented to tackle the independent task scheduling problem. In the algorithm, a task based representation is designed, and a new method is used to update the positions and velocity of particles. In order to keep the particle swarm algorithm from premature stagnation, the simulated annealing algorithm, which has local search ability, is combined with the PSO algorithm to make elaborate search near the optimal solution, then the quality of solutions is improved effectively. Experimental results compared with genetic algorithm and basic PSO algorithm show that the hybrid algorithm has good performance.

【Key words】 independent task scheduling; particle swarm algorithm; simulated annealing algorithm

1 概述

独立任务调度问题研究如何利用处理器资源最优地完成一批给定的任务(或作业)，通常以总完成时间最短为目标函数，是一类重要的组合优化问题，它不仅是并行与分布式计算中研究的关键问题，而且还广泛存在于工农业生产、交通运输及服务行业。

独立任务调度问题被证明是NP完全问题，即不存在多项式时间复杂性的算法以找到全局最优解。很多学者将启发式算法[1-2]、遗传算法、模拟退火算法和禁忌搜索算法应用到任务调度，构造了任务调度的遗传算法和一些混合算法，取得了满意的结果[3]。粒子群算法(Particle Swarm Optimization, PSO)[4]是由Kennedy和Eberhar于 1995 年提出的一种基于群智能的优化算法。该算法基于仿生学原理，通过模拟鸟群的觅食过程，在搜索过程中记忆个体最优和全局最优，使得种群中的所有粒子快速向最优解移动。由于其操作简单且易于实现，因此一经提出就受到广泛关注，目前已经在函数优化等领域得到了成功的应用。但对粒子群算法的研究目前主要集中在连续问题的优化求解方面，在离散组合优化方面的研究还非常有限，本文结合独立任务调度问题的特点，对传统PSO算法进行了改进，构造了一种基于任务编码的离散粒子群算法(Discrete Particle Swarm Optimization, DPSO)，将其与模拟退火(Simulated annealing, SA)算法结合，称之为DPSO-SA。

2 问题描述

独立任务调度问题通常用 $Pm\|C\max$ 表示，具体描述如下：设有n个独立任务 $T_1,T_2,\cdots,T_n$ 需要竞争使m台机器 $P_1,P_2,\cdots,P_n$，任一任务可以在任何处理机上运行，未完工的任务不允许中断，任务也不能拆分成更小的子任务，分配的目标是把这n个任务合理地分配到m台机器上执行，使总的执行时间最短。用一个n维向量 $(t_1,t_2,\cdots,t_n)$ 表示任务的估计执行时间，其中元素t_i为任务i的估计执行时间；定义处理机执行时间为处理机完成分配给它的所有任务所需的时间，其中最大的处理机执行时间称为调度长度。假设有一个调度策略，它为处理机P_i上分配 k_i 个任务 $T_{i1},T_{i2},\cdots,T_{ik_i}$，则处理机 P_i 的完成时间 $C_i=t_{i1}+t_{i2}+\cdots+t_{ik_i}$。独立任务调度的数学模型为

$$\min(\max_{1\le i\le m}C_i)=\min(\max_{1\le i\le m}\sum_{j=1}^{k_i}t_{ij})$$

$$\text{s.t.}\quad t_{ij}\ge 0$$

$$\sum_{i=1}^{m}k_i=n$$

3 离散粒子群优化算法

3.1 基本粒子群算法

基本粒子群算法采用速度-位置模型进行搜索，待优化问题的每个候选解称为一个粒子，它具有位置和速度两个特征，算法首先随机初始化产生一个粒子群体，然后进行迭代寻优。每一次迭代中，粒子通过跟踪两个“极值”来更新自己：一个是粒子本身找到的最优解，称为个体极值 *pbest*，另一个是整个粒子群目前找到最优解，称为全局极值 *gbest*，粒子的位置和速度更新如式(1)、式(2)所示。

$$V_i=wV_i+c_1Rand()(pbest_i-X_i)+c_2Rand()(gbest-X_i) \quad (1)$$

基金项目：山东省自然科学基金资助项目(2004ZX14)；聊城大学自然科学基金资助项目(X051033)

作者简介：陈 晶(1975－)，女，讲师、硕士，主研方向：智能优化算法；潘全科，教授、博士

收稿日期：2007-03-30 **E-mail：**lcucj@163.com

$$X_i = X_i + V_i \tag{2}$$

其中，V_i 为粒子的速度；X_i 为粒子的位置；$pbest_i$和$gbest$为个体极值和全局极值；$Rand()$ 为区间[0,1]上的随机数；w为惯性权重，其主要作用是产生扰动，防止算法的早熟收敛，一般在 0.1~0.9 之间取值，c_1和c_2为学习因子，调节向个体最优粒子和全局最优粒子方向飞行的最大步长，一般取c_1=c_2=2。迭代过程中，粒子的速度和位置都限制在特定的范围内，同时$pbest_i$和$gbest$不断更新，最后输出的$gbest$就是全局最优解[4]。

3.2 改进的 DPSO 算法

在求解组合优化问题时，由于当前粒子与最优粒子位置难以用定量的方式表示，导致粒子的速度和位置不能参照式(1)、式(2)进行更新，此时基本 PSO 算法失效。为使 PSO 算法能够在离散组合优化问题中得到有效应用，文献[5]提出了二进制编码的 PSO 算法，把位置表示成离散型的 0 和 1，文献[3]提出了求解旅行商问题的 DPSO 算法，但其时间性能及仿真效果都不理想。文献[6]以遗传算法为基础，采用二进制编码的粒子群算法求解独立任务调度问题，获得了较好的求解质量，但其编码较长，算法运行效率不高。结合 SA 算法，本文设计了用于求解独立任务调度的 DPSO 算法，仿真结果表明，与遗传算法(Genetic Algorithm, GA)和基本粒子群算法相比，该算法是一种可行而高效的算法。

设计 DPSO 算法的关键是将粒子的位置、速度及其运算规则与特定问题域结合，用恰当的方式表示出来。本文设计了一种基于任务的自然数编码的粒子群算法，编码长度取决于任务数量，例如对于 n 个任务 m 台处理机的调度问题，用 n 维向量 $(d_1,d_2,\cdots,d_n)$ 表示粒子的位置，向量的第 i 维分量表示第 i 个任务分配到第 $d_i(1\le d_i\le m)$ 台处理机上执行。

例如 6 个任务 3 台处理机的调度问题，表 1 给出了一个可行的调度方案。如任务、位置分量对(2, 1)表示把任务 2 分配到处理器 1 上执行。

表 1　位置矢量及对应的任务调度分配方案

任务	位置向量$\boldsymbol{X}_i$
1	3
2	1
3	2
4	2
5	1
6	2

粒子群算法的实质在于粒子根据自身的飞行经历和同伴的飞行经历进行位置和速度的动态调整，从而向最优位置飞行。结合独立任务调度问题的特点，本文重新定义了位置更新公式。为避免速度之间的互相干扰，在式(3)中采用了分步计算和修改粒子位置的方式，首先将粒子自身的最佳飞行位置$pbest_i$作用于当前位置上，然后根据群体最佳位置$gbest$对当前粒子进行调整。

$$\begin{cases}\boldsymbol{X}_i = c_1 \times g(w\times f_{a,b}(\boldsymbol{X}_i), pbest_i)\\ \boldsymbol{X}_i = c_2 \times g(\boldsymbol{X}_i, gbest)\end{cases} \tag{3}$$

其中，w, c_1, c_2的取值均在区间[0, 1]内；$f_{a,b}(\boldsymbol{X}_i)$ 是关于位置 $\boldsymbol{X}_i$ 的函数，其作用是交换 $\boldsymbol{X}_i$ 的第a个分量与第b个分量(a, b为 0 到n之间的随机整数)。为保持粒子群的多样性，使算法在迭代后期依然能够保持进化能力，本算法以 w 为概率对粒子位置进行扰动，式中 $w\times f_{a,b}(V_i)$ 的含义为

$$w\times f_{a,b}(\boldsymbol{X}_i) = \begin{cases} f_{a,b}(\boldsymbol{X}_i) & \text{if} \quad \rho < w \\ \boldsymbol{X}_{ii} & \text{else}\end{cases} \tag{4}$$

其中，$\rho \in [0,1]$。

$g(X_i, pbest_i)$ 为 $\boldsymbol{X}_i$ 对$pbest_i$的学习操作，具体过程如下：首先把 m 台处理机随机分成两个集合 S_1 和 S_2，对于 $\boldsymbol{X}_i$ 和$pbest_i$的第j维分量，令

$$g(x_{i,j}, pbest_{i,j}) = \begin{cases} x_{i,j} & \text{if} \quad x_{i,j}\in S_1 \\ pbest_{i,j} & \text{else}\end{cases} \tag{5}$$

$g(x_{i,j}, gbest_j)$ 的定义同式(5)。

基于任务编码的DPSO算法编码简单，计算量较少，算法的优化性能得到了有效的改善。DPSO算法在迭代初期收敛速度较快，但是随着个体极值和全局极值的作用，粒子很快会失去多样性，算法丧失全局搜索能力，为解决这一问题，本文引入SA算法。SA算法是一种求解大规模组合优化问题，特别是NP完全问题的有效近似算法。它源于对固体退火过程的模拟，采用Metropolis接受准则，并用一种称为冷却进度表的参数控制算法进程，使算法在多项式时间里给出一个近似最优解[7]。由于SA算法能够以一定的概率接受质量差的解，从而避免陷入局部最优，因此将DPSO与SA相结合产生一种新的混合算法，算法的执行分为两步：(1)利用PSO算法的快速搜索能力获得一个较好的群体；(2)利用SA算法进行精细搜索。

具体算法流程如下：

(1)初始化

1)PSO参数：确定w, c_1, c_2, λ(λ为惯性权重下降速率)和群体规模，设定最大世代数 $Maxgen$，随机初始化粒子群中粒子的位置，并评价粒子的适应度。令$gbest_i=X_i$, $gbest$设置为初始群体中最佳粒子的位置。

2)SA 参数：确定初温 t_0，退温系数 β。

(2)执行 PSO 算法

当前世代数< $Maxgen$ 时，

{ 按式(3)更新粒子位置，并计算适应度。

更新种群的 $gbest$ 和每个粒子的 $pbest$，

令 $w=\lambda w$，$k++$，转(4)。}

(3)执行模拟退火算法

{ 令 i=0；$t=t_0$；

判断算法收敛准则是否满足，如满足，输出 $gbest$，得到优化结果，算法结束，否则

{ 产生 S 的一个邻域解 S'，计算新解的适应值；

如果 S' 优于 $gbest$，则令 $gbest$= S'；否则按照接收准则接收新状态。}

以退火温度收敛率 β 逐步降低温度，即 $t=\beta t$ }

4 仿真实验及结果

以 VC＋＋作为开发环境，采用与文献[6]相同的实验数据，验证了算法的有效性。设有 3 台处理机和 9 个作业，作业需要运行时间分别为：81s, 40s, 26s, 4s, 65s, 98s, 53s, 71s, 15s。采用遗传算法、DPSO 算法及合算法 DPSO-SA 分别测试其调度性能。

DPSO算法的参数设置为：粒子数为 30，w=0.9, c_1=0.9, c_2=0.9, λ=0.975；SA算法的参数设置为：t_0=100, β=0.96，最大迭代次数为 50 次。各种算法均重复测试 100 次，统计出最好解、最差解、最好解的次数和平均解。表 2 给出了仿真实验的结果。表中第 2 行数据选取的是文献[6]的最优结果。比较可知，采用基于任务编码的DPSO-SA算法在求解任务调度问题时性能比较稳定，几乎每次都能找到最优解，其最好解出现的次数优于GA, DPSO算法及文献[6]中的算法。

(下转第 218 页)