基于混合粒子群算法的网格任务调度

王成昌 陈闵中 方 钰 邓 蓉
(同济大学电子与信息工程学院 上海 201804)

摘 要 减少分布式程序的执行时间是网格调度系统需要解决的重要问题。因分布式程序常建模为DAG图,故该问题又称异构DAG调度问题。在研究网格环境下的任务调度的基础上,提出了一种用于解决DAG任务调度问题的通用混合粒子群优化算法(Common Hybrid Particle Swarm Optimization),简称为CHPSO。该算法将问题的解(粒子)表示为任务的调度优先权向量,采用混合粒子群优化算法探索解空间。实验结果表明,在求解不含孤立点的单个DAG调度问题时,该算法所得解的调度长度仅为HEFT的90%~92%,求解质量与PSGA相当;在多张DAG图(含孤立节点)并发执行的网格环境中,该算法的调度性能明显优于PSGA及文中列出的其它演化计算方法。

关键词 网格,DAG调度,粒子群优化算法

Task Scheduling in Grid Environment Based on Hybrid PSO Algorithm

WANG Cheng-chang CHEN Hong-zhong FANG Yu DENG Rong
(Department of Computer Science and Technology, Tongji University, Shanghai 201804, China)

Abstract Reducing execution time of distributed program is a major issue of grid scheduling system. Because scheduled programs are modeled by DAG, this problem is called Heterogeneous DAG scheduling problem also. Based on the research of task scheduling in grid environment, an algorithm named common hybrid particle swarm optimization(CHPSO) was proposed to solve the DAG scheduling problem. The algorithm presents the solution of the problem(particles) as a priority vector of the scheduling task and utilizes the hybrid PSO algorithm to explore solution space. Experimental result indicates that, in pure DAG scheduling which has no isolate task node, the CHPSO can get a scheduling length only 90%~92% of HEFT algorithm and as good as PSGA, but in grid environment where multi DAG graphs are concurrently executed, this algorithm performs obviously better than PSGA and other evolutionary computation listed in this paper.

Keywords Grid, DAG scheduling, Particle swarm optimization

网格任务调度是网格系统中重要的组成部分,它的主要功能可以简单地描述为根据任务信息和当前资源信息,采用适当的策略为任务选择最合适的资源。在网格系统中运行着许多科学工作流或商业工作流,通常人们用有向无环图(Directed Acyclic Graph,简称为DAG)为分布式程序建模[14]。为了满足程序的性能要求,网格必须提供足够的资源管理和调度服务来有效解决异构DAG图调度问题。网格任务调度以缩短整个程序的运行时间为优化目标,并且已被证明为NP完全的组合优化问题,不存在多项式时间复杂度的解,而启发式算法在解决这类问题上有着明显优势。过去的十几年,研究者们从人类日常生活经验和生物群落行为中汲取灵感,针对此问题提出了许多启发式算法[2,4]和演化计算方法[5-7]。虽未能完满解决所研究的问题,但这些算法为后续研究奠定了坚实的基础。

粒子群优化算法(Particle Swarm Optimization,PSO)是一种演化计算技术(Evolutionary Computation),由Eberhart博士和Kennedy博士在文献[9]中首次提出,并应用于连续非线性函数的优化求解。PSO算法源于对鸟群捕食的行为研究,同遗传算法类似,是一种基于迭代的优化工具。系统初始化为一组随机解,通过迭代搜寻最优值。PSO基本不受问题峰数增加的影响,其受问题维数的影响也很小,在连续多维空间多峰问题寻优、动态目标寻优等方面有着速度快、解质量高、鲁棒性好的优点,因此已广泛应用于函数优化、神经网络训练、模糊系统控制以及其他遗传算法的应用领域。

文中尝试用PSO方法求解异构系统中的多张DAG图并发调度问题,得到了算法CHPSO(Common Hybrid Particle Swarm Optimization,通用混合粒子群优化算法)。该算法将粒子编码为表示任务调度优先数的实数向量,成功地将离散优化领域的异构DAG调度问题转化为适合PSO求解的实数空间连续优化问题。其次,为保证粒子的合法性,不会出现前驱任务优先级高于后继的情况,即算法对任务的优先数限界;但分属不同DAG图的任务彼此独立,CHPSO随机生成任务

到稿日期:2011-04-05 返修日期:2011-07-21 本文受国防基础研究计划基金项目(A1420080182)资助。

王成昌(1987-),男,硕士,主要研究方向为操作系统、分布式计算等,E-mail:wangchang517@163.com;陈闵中(1951-),男,博士生导师,主要研究方向为操作系统、分布式计算等;方 钰(1977-),女,副教授,主要研究方向为操作系统、网格计算;邓 蓉(1973-),女,讲师,主要研究方向为操作系统、分布式计算等。

优先数，确保不存在前驱后继关系的多个任务调度次序任意。最后，CHPSO 引入变异操作，对基本 PSO 位置更新操作后的新粒子进行随机扰动，以均衡算法的全局搜索能力和局部寻优能力，改善种群的多样性，克服早熟现象，以避免陷入局部最优。实验结果表明，该算法性能优异，求解单 DAG 调度问题时，解的调度长度仅为 HEFT 的 90%～92%，求解质量与 PSGA 相当；在多张 DAG 图（含孤立节点）并发执行的网格环境中，该算法的调度性能明显优于 PSGA 及本文列出的其它演化计算方法。

本文第 1 节对网格任务调度问题进行形式化描述；第 2 节介绍本文算法实现；第 3 节对实验结果进行分析；最后总结全文。

1 问题描述

解决网格环境下的分布式程序调度，又称异构 DAG 调度问题，是网格资源管理必须解决的重要问题。为了缩短整个程序的运行时间（或对开销等其它指标进行优化），任务调度管理器需要为程序所包含的所有任务分配处理器资源并优化它们的执行时间。

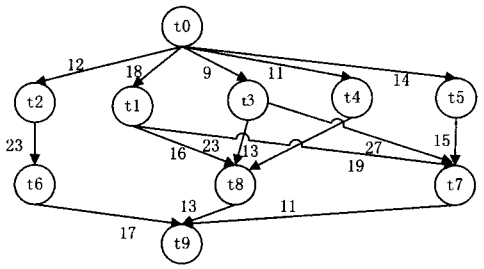

图 1 HEFT 算法中经典 DAG 用例图[11]

表 1 符号和定义

符号	定义
prec(i)	任务 t_i 的直接前驱集合
Succ(i)	任务 t_i 的直接后继集合
Proc(i)	分配给任务 t_i 的处理器
Ch	当前高度
PNUM	粒子群中粒子个数
AvailableTC	可分配任务集合
DagNums	所有 DAG 图中子任务总个数
SPAN	任务优先级跨度
R_i，M	编号为 i 的资源，M 表示资源个数
readyTime(R_i)	资源 R_i 的就绪时间
ps_i	粒子，编号为 I，包含任务执行序列，调度长度，适应度，任务资源映射方案
SL(ps_i)	粒子 ps_i（编号为 i 的粒子）的 makespan，即调度长度
Fitness(ps_i)	粒子 ps_i 的适应度
Random(num)	0～num 范围内的随机数
Random(m,n)	m～n 之间的随机数
Prior[i,t_i]	第 i 个粒子中任务 t_i 对应的优先权值
compTime(t_i)	任务 t_i 的完成时间
readyTime(R_i)	资源 R_i 的就绪时间
pBest(i)	粒子编号为 i 的个体最优粒子
pBest(I,j)	pBest(i) 中任务编号为 j 的优先权值
gBest	在历代进化过程中整个粒子群中所发现的最优粒子
gBest(j)	gBest 中任务编号为 j 的优先权值
C(i,j)	任务 t_i 和 t_j 之间数据传输所需的时间
ETC(i,j)	Estimated Time to Complete，任务 t_i 在处理器 R_j 上估计执行时间
C1，C2	学习因子
P[I,j]	ps_i 中任务编号为 j 的优先权值
P′[I,j]	p[I,j]执行减法操作后的优先权值
MRATE	粒子优先权序列中优先权值的变异概率
P″[I,j]	P′[I,j]执行编译操作后的优先权值
ITER	粒子寻优过程中迭代次数

将被调度程序建模为有向无环图 $\Omega(T,E)$：任务集合 $T=\{t_i\}$，$i\in[0,N-1]$，N 表示子任务个数；边集 $E=\{e_{i,j}\}$，$i,j\in[0,N-1]$，其中 $e_{i,j}$ 表示任务 t_i 和 t_j 之间的数据依赖关系：t_i 是 t_j 的直接前驱，t_j 是 t_i 的直接后继，边的权重表示任务 t_i 运行结束后向 t_j 传送的数据量。任务 t_j 必须在接收到其所有前驱的数据后才可以运行，当任务 t_i 和 t_j 分布在同一资源节点上时，数据传输时间为 0。本文中以经典 HEFT 算法给出的 DAG 图为例，其任务图描述如图 1 所示。

程序的运行环境为一组全连接的处理器集合（连接在 Internet 上的每台处理器均可直接与其它处理器通信，故该假设适用于计算网格）。任务可以在所有处理器上运行，其执行代价用矩阵 ETC 表示。每台处理器空分、非剥夺，也就是说一旦任务开始在某处理器上开始执行，将独占该处理器直至运行结束。表 1 所列的是本文所使用的记号及它们的含义。

2 算法描述

使用粒子群算法解决该问题时，每一个粒子都应该直接或间接对应着问题的一个合法解。当给定任务集合中的任务描述和可用资源时，通过设定任务执行的优先级，按优先权大小进行非递减排列即可以得到一个任务执行序列，通过合适的分配规则依次对该任务序列中的任务进行资源分配，可得到任务最终执行完成的时间，该完成时间即对应任务调度问题的一个解。如何获取最优的任务执行序列，是解决该问题的关键。在本文的混合粒子群算法中，一个粒子即对应着一个优先权序列，所采用的任务分配规则为 EFT（最早完成时间）。整个算法流程可简单描述为在初始化粒子群并给定粒子适应度的计算方法后，通过迭代使用解空间搜索策略，找寻最优粒子，从而获取问题近似最优解的过程。CHPSO 算法的基本步骤如下：

Step 1 初始化算法参数：设定惯性系数、认知系数、种群规模和结束条件。

Step 2 初始化粒子群、个体极值和全局极值。

Step 3 根据式(4)和式(5)更新粒子的位置，并依据式(6)给出的约束，调整粒子位置。

Step 4 计算每个粒子的适应值，使用式(8)判定粒子质量并确定新粒子的取舍。

Step 5 评价种群：

Step 5.1 如果当前粒子的适应值优于其个体极值，则更新个体最优位置和个体极值。

Step 5.2 如果当前群体中的最优值优于全局极值，则更新全局最优位置和全局极值。

Step 6 如果达到结束条件，则输出全局极值，算法结束；否则转 Step 3。

2.1 初始化粒子群

粒子群中每个粒子包含一个使用 n 维向量表示的优先权序列，同时存储有粒子对应的调度长度、任务分配方案以及适应度大小。初始化包含 PNUM 个粒子的粒子群，第 i 个粒子的初始化方法如下：

Step 1 定义当前高度 $ch=0$，初始化任务集合 T 边集 E，从 T 中选取无前驱节点的任务（包括独立任务）放入可分配任务集合

$AvailableTC=\{t_i, prec(i)=\emptyset\}$

Step 2　若可分配任务集合中存在独立任务节点(既无前驱又无后继节点),则首先对所有独立任务的优先权进行初始化,独立任务 j 的优先权生成规则如下:

$prior[i,t_i]=random(dagNums*SPAN)$,

然后将对应任务从可分配任务集合中移除。若无,则跳过此步。

Step 3　对当前可分配任务集合中任务生成优先权,规则如下:

$prior[i,t_i]=ch*SPAN+random(SPAN)\quad(t_j\in AvailableTC)$

Step 4　从当前可分配任务集合中选择优先权值最小的任务,并记录其对应的优先权值,然后将该任务从可分配任务集合中移除,当前高度值 $ch=ch+1$,并标记该任务为"已分配"。此时检查该任务的所有后继任务节点是否可分配(若该后继任务的所有前驱节点都已分配,则将该任务加入到可分配任务列表);若当前可分配任务集合为空,则该粒子优先权初始化完毕;否则,继续 Step 3,重新计算 TC 集合中所有任务的优先数。

在粒子初始化过程中,使得独立任务的执行顺序具有任意性,DAG 任务满足其前驱后继约束。对于 DAG 任务中可并发执行的子任务,采用随机优先权的方式来确定任务间执行顺序。示例:对图 1,取 $SPAN=100$,采用上述方法生成的一个粒子其优先权值如表 2 所列。

表 2　图 1 中有效粒子优先权示例

t0	t1	t2	t3	t4	t5	t6	t7	t8	t9
2	223	302	111	405	744	511	892	614	938

2.2　计算粒子适应度

对粒子群中每个粒子,其适应度计算方法如下:

Step 1　对粒子中优先权序列进行非递减排序,记录每个任务排序后在序列中所对应的位置,即得到一个有效的任务执行序列。对表 2 所列粒子,其对应的有效任务执行序列即为[t0,t3,t1,t2,t4,t6,t8,t5,t7,t9]。

Step 2　初始化所有可用资源的就绪时间 $readyTime=0$,所有任务完成时间 $compTime(t_i)=0$,依次对任务执行序列中任务 $t_i(0\leqslant i<N)$ 分配资源。

Step 3　对所有资源节点,尝试分配任务 t_i,假设当前资源为 R_j,首先计算其前驱节点约束的最早开始时间 $EST(t_i)$,计算方法如下:

$$EST(t_i)=\max(compTime(t_j)+c[j][i])\,t_j\in prec(t_i) \tag{1}$$

若 t_j 也分配在资源 R_j 上时,$c[j][i]=0$,然后计算任务 t_i 分配在编号为 j 的资源上的最早完成时间 $EFT(t_i,R_j)$,计算方法如下:

$$EFT(t_i,R_j)=\max(EST(t_i),readyTime(R_j))+ETC[i][j]$$

$$prior[i,t_i]=ch*SPAN+random(SPAN)\quad(t_j\in AvailableTC) \tag{2}$$

Step 4　将任务分配到拥有最早完成时间的资源上,更新该任务的完成时间 $compTime(t_i)=\min\{EFT(t_i,R_j)\ 0\leqslant j<M\}$ 及对应资源的就绪时间 $readyTime(R_j)=compTime(t_i)$。若所有任务已全部分配到相应资源,则该粒子对应的执行时间即为所有资源的就绪时间的最大值,否则,继续 Step3。

示例:对表 1 所列粒子,采用 EFT 算法进行资源分配,其资源分配结果如表 3 所列,对应的执行时间为 84。

表 3　表 1 中粒子对应资源分配方案

任务	t0	t1	t2	t3	t4	t5	t6	t7	t8	t9
资源	R2	R2	R0	R1	R2	R2	R0	R0	R1	R1

得到当前粒子群中所有粒子的执行时间后,计算每个粒子对应的适应度如下:

$$fitness(ps_i)=1/SL(ps_i)\Big/\sum_{i=1}^{pNum}(1/SL(ps_i)) \tag{3}$$

2.3　混合粒子群算法的思想搜索解空间

对粒子群每个粒子,根据 SL 更新其个体最优 $pBest$ 以及全局最优个体 $gBest$。对于粒子 ps_i 优先权序列中的每一位,使用基本粒子群算法中的减法操作对其进行更新。对优先权序列中的第 j 位,即任务 t_j 的优先权,其更新方式如式(4)所示。

$$p'[i,j]=p[i,j]+C1*random()*(pBest[i,j]-p[i,j])+C2*random()*(gBest[j]-p[i,j]) \tag{4}$$

对更新后的优先权 $P'[i,j]$,按变异概率 $mRate$ 选择是否要进行变异操作。对优先权序列中第 j 位进行变异的方式如式(5)所示。

$$p''[i,j]=p'[i,j]+random(-span,span) \tag{5}$$

对交叉变异后得到的新优先权,需要检查任务 t_j 对应优先权值是否越界(即 t_j 优先权值应小于其后记节点中最小的优先权且大于其前驱结点集中最大的优先权),任务 j 的优先权范围的符号描述如式(6)所示。

$$bound\_down(t_j)=\begin{cases}\max\limits_{t_i\in prec(t_j)}(prior[t_i]), & if(prec(t_j)\neq\emptyset)\\ 0, & else\end{cases}$$

$$bound\_up(t_j)=\begin{cases}\min\limits_{t_i\in succ(t_j)}(prior[t_i]), & if(succ(t_j)\neq\emptyset)\\ dagNums*span, & else\end{cases} \tag{6}$$

式中,$bound\_down(t_i)$ 表示当前粒子中任务 t_j 的优先权下界,$bound\_up(t_i)$ 表示任务 t_j 的优先权上界。若优先权值越界,即 $p''[i,j]<=bound\_down(t_j)\ ||\ p''[i,j]>=bound\_up(t_j)$,则对 $P''[j]$ 反复做如下调整:

若 $P''[i,j]>=bound\_up(t_j)$ 则 $P''[i,j]=p[i,j]+(P''[i,j]-boudn\_up(t_j))$

若 $P''[i,j]<=bound\_down(t_j)$ 则 $P''[i,j]=p[i,j]+(P''[i,j]-bound\_down(t_j)$

如果调整中发现 $P''[j]=bound\_down(t_j)$,则令 $P''[j]=P''[j]+\Delta$

同理,若 $P''[j]=bound\_up(t_j)$,则令 $P''[j]=P''[j]-\Delta$,其中 $\Delta=0.01$。

优先权调整操作,避免了 DAG 任务中出现节点先于其前驱节点或后于其后继节点执行的情况。对最终得到的新粒子首先使用 2.2 节中方法计算其执行完成时间 $newSL$,若新粒子的调度时间优于原始粒子,则以新粒子替换原始粒子;否则通过式(7)计算新粒子对应的适应度大小。

$$fitness'(ps_i)=[SL(ps_i)*fitness(ps_i)]/newSL \tag{7}$$

然后计算新旧粒子变化范围:

$$\Delta fitness=fitness(ps_i)-fitness'(ps_i) \tag{8}$$

若 $\Delta fitness<\Delta E$(ΔE 为按允许目标变坏范围[12]),则替

换原始粒子，这里取 $\Delta E=1/(10*PNUM)$，否则，保留原始粒子。

3 实验与分析

为验证本文算法（即 CHPSO）的有效性，统一选取 $PNUM=20$，$ITER=1000$，$C1=C2=1.5$，$SPAN=100$，$MRATE=0.3$ 并针对以下两种场景进行实验分析：

1. 首先使用 CHPSO 算法对无孤立节点的纯 DAG 图描述的单个分布式任务进行调度。在测试用例中，对问题描述中给出的经典 HEFT 图（表 4 中 classicHEFT. stg）进行测试，同时还引入了文献[13]中的 DAG 图（表 4 中 example1. stg）以及文献[3]中的 DAG 图（表 4 中 example2. stg）。

就笔者所知，目前没有异构 DAG 调度问题的标准测试集，因此没有最优解可供参照。因 CHPSO 嵌入了 HEFT 使用的启发式知识，使用 HEFT 作为基准比较其它算法的性能。分别使用 CHPSO，HEFT，PSGA[8] 算法对上表中每个 DAG 用例运行 10 次，并记录 10 次运行结果中最优解和平均值。实验结果如表 4 所列。

表 4 对纯 DAG 图的算法调度结果

	任务数	资源数	CHPSO		PSGA		HEFT
			最优	平均	最优	平均	
classic. stg	10	3	73	74	73	75	80
example1. stg	7	3	2018	2018	2018	2043	2069
example2. stg	9	4	143	143	143	143	149
antiHEFT. stg	6	2	12	12	12	12	13
test4. stg	8	2	84	85	84	91	93
new. stg	11	3	73	73	73	74.5	80

从表 4 可以看出，CHPSO 算法得到的最优解和平均值明显优于 HEFT。而且 CHPSO 得到的最优解虽相比 PSGA 无明显优势，但平均值优于 PSGA 算法，即 CHPSO 得到最优解次数较多。

2. 第二组实验针对求解混合任务类型做分析，所谓混合任务类型就是独立任务和 DAG 任务并存，或多个 DAG 任务并存的情形。表 5 中 integ1. stg 为表 4 中 classic. stg 和 new. stg 的合集，integ2. stg 为表 4 中 antiHEFT. stg 和 test4. stg 的合集，其余 3 个任务图在则表 4 中 DAG 图的基础上，添加 3～5 个随机独立任务节点，或者添加独立任务节点和部分 DAG 任务节点，然后分别使用 PSGA，CHPSO，HEFT 进行调度，调度结果如表 5 所列。

表 5 混合任务类型的算法调度结果

	任务数	资源数	CHPSO		PSGA		HEFT
			最优	平均	最优	平均	
integ1. stg	21	3	88	91	91	93	104
integ2. stg	14	2	91	91	92	93.2	93
add1. stg	14	4	143	143.7	144	148.4	151
test4a. stg	10	2	117	117	119	121	124
multi. stg	20	3	133	134.2	142	146.8	152

从实验结果可以看出，在对混合任务类型的调度问题上，CHPSO 的性能无论在最优解还是平均值上均优于 PSGA 和 HEFT 算法。而且从平均值和最优解之间差值可以看出，CHPSO 的稳定性较好。

结束语 本文给出通用混合粒子群算法 CHPSO，用以解决网格环境下的任务调度问题。因恰当地使用了启发式知识，该算法在调度纯 DAG 任务时表现出的性能明显优于 HEFT 算法，且不差于 PSGA。值得注意的是，本文采用的粒子编码方式，可使该算法支持混合任务类型的调度。而且在混合任务类型调度中，CHPSO 算法在性能和稳定性上都明显优于 PSGA 和 HEFT。在真实网格环境中，多 DAG 任务的并行调度不可避免，从而该算法更适应于网格环境下的任务调度。

参 考 文 献

[1] UllMan J D. NP-complete scheduling problems[J]. J. of Computer and Systems Sciences, 1975, 10: 384-393

[2] Topcuoglu H, Hariri S, Wu Min-you. Performance effective and low-complexity task scheduling for heterogeneous computing [J]. J. of IEEE Transactions on Parallel and Distributed Systems, 2002, 13(3): 260-274

[3] Kwok Y K, Ahmad I. Link contention-constrained scheduling and mapping of tasks and messages to a network of heterogeneous processors[J]. Cluster Computing, 2000, 3(2): 113-124

[4] Cirou B, Jeannot E. Triplet: A clustering scheduling algorithm for heterogeneous systems [C] // International Conference on Parallel Processing Workshop. 2001: 31-236

[5] Hou E S, Ansari N, Ren H. A Genetic Algorithm for Multiprocessor Scheduling [J]. J. of IEEE Transactions on Parallel and Distributed Systems, 1994, 5(2): 113-120

[6] Kennedy J, Eberhart R C. Particle Swarm Optimization [C] // Proceedings' of the Fourth IEEE International Conference on Neural Networks. Perth, Australian: IEEE Press, 1995, 4: 1942-1948

[7] Correa R C, Ferreira A, Rebreyend P. Scheduling Multiprocessor Tasks with Genetic Algorithms [J]. J. of IEEE Transactions on Parallel and Distributed Systems, 1999, 10(8): 825-837

[8] Dhodhi M K. An integerated Technique for Task Matching and Scheduling onto Distributed Heterogeneous Computing Systems [J]. Journal of Parallel and Distributed Computing, 2002, 62: 1338-1361

[9] Kennedy J, Eberhart R. Particle swarm optimization [J]. J. of IEEE International Conference on Neural Networks, proceedings, 1995, 4: 1942-1948

[10] Deng Rong, Jiang Chang-jun, Yin Fei. Ant Colony Optimization for Precedence-constrained Heterogeneous Multiprocessor Assignment Problem[C]//GEC'09 Proceedings of the first ACM/SIGEVO Summit on Genetic and Evolutionary Computation, 2009

[11] Topcuoqlu H, Hariri S, Wu M Y. Performance-effective and lw-complexity task scheduling for heterogeneous computing [J]. IEEE Transactions on Parallel and Distributed Systems, 2002, 13(3): 260-274

[12] 高尚，杨静宇. 群智能算法及其应用[M]. 北京：水利水电出版社，2006

[13] Watson S P, Flann D W N S, et al. Genetic simulated annealing for scheduling data dependent tasks in heterogeneous environments [C] // Proceedings of the Heterogeneous Computing Workshop. 1996: 98-104

[14] 邓蓉，陈闵中，王博，等. 一种求解异构 DAG 调度问题的置换蚁群[J]. 计算机科学，2010 37(12): 193-196

[15] Kang Qin-ma, He Hong. A Novel Discrete Particle Swarm Optimization Algorithm for Meta-task Assignment in Heterogeneous Computing Systems [J]. Microprocessors and Microsystems, 2011, 35(1): 10-17