Journal of Computer Applications
计算机应用,2014,34(6):1798 - 1802
ISSN 1001-9081
CODEN JYIIDU
2014-06-10
http://www.joca.cn

文章编号:1001-9081(2014)06-1798-05
doi:10.11772/j.issn.1001-9081.2014.06.1798

协同震荡搜索混沌粒子群求解资源受限项目调度问题

戴月明,汤继涛*,纪志成
(江南大学 物联网工程学院,江苏 无锡 214122)
(*通信作者电子邮箱 tangjitao8723@126.com)

摘　要:针对求解资源受限项目调度问题(RCPSP),提出了协同震荡搜索混沌粒子群(CSCPSO)算法。算法围绕种群粒子吸引子建立双向协同震荡搜索机制,该机制一方面使粒子向吸引子收敛,另一方面使粒子震荡调整自身与吸引子相邻维度大小关系不一致的维度,提升算法的搜索精度和种群的多样性。项目调度采用基于粒子的拓扑排序和串行项目进度生成机制,保证项目调度解决方案满足资源约束和紧前约束。采用具体算例对算法进行检验,结果表明该算法在求解 RCPSP 的精度和稳定性方面表现更优。

关键词:协同震荡搜索;混沌;粒子群优化算法;拓扑排序;资源受限项目调度问题
中图分类号:TP391　　**文献标志码**:A

Cooperative shock search particle swarm optimization with chaos for resource-constrained project scheduling problems

DAI Yueming, TANG Jitao*, JI Zhicheng
(*School of Internet of Things Engineering, Jiangnan University, Wuxi Jiangsu* 214122, *China*)

Abstract: For Resource-Constrained Project Scheduling Problems (RCPSP), the Cooperative Shock search Particle Swarm Optimization with Chaos (CSCPSO) was proposed. On the basis of particle attractor, a bidirectional cooperation shock search mechanism was established in the algorithm to enhance the search accuracy and diversity of population. The particles converged to particle attractor, meanwhile they adjusted the dimensions whose adjacent relationship were inconsistent with attractor's by shock search in the mechanism. Combined with topological sorting based on particles and serial schedule generation scheme, the gotten scheduling scheme could meet the project schedule constraints of resource and precedence relations. The tests on specific examples show that the proposed algorithm can get higher accuracy and better stability for RCPSP.

Key words: cooperative shock search; chaos; Particle Swarm Optimization (PSO) algorithm; topological sorting; Resource-Constrained Project Scheduling Problem (RCPSP)

《计算机应用》Journal of Computer Applications

0　引言

资源受限的项目调度问题(Resource-Constrained Project Scheduling Problem, RCPSP),是在满足活动时序、资源约束等条件下安排各活动开始和结束时间以达到工期最小的调度方法[1]。RCPSP 广泛存在于各个行业中,该问题已被证明是 NP-hard 问题[2]。因其求解困难,吸引着国内外众多学者研究,迄今已提出了一系列的问题模型,形成了一套比较完善的模型库。RCPSP 优化方法概括起来可分为精确算法和启发式算法。精确算法用于求解小规模调度问题,但对大规模的问题求解力不从心。对于大型复杂的调度问题,引入了各类启发式算法,如遗传算法、蚁群算法、模拟退火算法[3-5]等。目前,一种在其他领域应用比较成功的群智能算法——粒子群优化(Particle Swarm Optimization, PSO)算法也被引入来解决这类问题[6]。然而,原始的 PSO 在求解项目解决方案时,由于其自身易陷入局部最优值的缺点,无法准确获取项目的最佳解决方案。本文针对 PSO 的局限性和项目的机动性,提出协同震荡搜索混沌粒子群(Cooperative Shock search Particle Swarm Optimization with Chaos, CSCPSO)算法,动态自适应地协同双向调整搜索策略,使粒子减小陷入局部最优的概率,提升在搜索空间内获取最佳解决方案的精度,并通过实际算例验证算法的有效性。

1　模型描述

对于经典 RCPSP 模型,一般有以下两个假设:

1) 只考虑 $FS = 0$ 的逻辑关系,即所有紧前任务结束后,后续任务可以立即开始,不存在延迟现象。

2) 只考虑可更新资源,即资源在每个时刻的供应量是有限的,但不会随着项目的进展而产生损耗。

基于前述的两个假设,RCPSP 可以描述为:一个项目包含 D 项任务数,项目所有的任务集合 $J = \{1,2,3,\cdots,D\}$,d_j 为任务 j 的工期,项目的整个工期为 T;P_j 表示任务 j 的紧前任务集合,任务 j 在其任一紧前任务 $p(p \in P_j)$ 完成前不能开始,S_j 表示任务 j 的紧后任务集合;项目共有 K 种可更新资源,第 k

收稿日期:2013-10-21;**修回日期**:2014-04-13。
基金项目:国家 863 计划项目(2013AA040405);江苏省产学研联合创新基金资助项目(BY2012055)。
作者简介:戴月明(1964 -),男,江苏常熟人,副教授,主要研究方向:人工智能、模式识别、数据挖掘;　汤继涛(1987 -),男,江苏邳州人,硕士研究生,主要研究方向:人工智能、模式识别;　纪志成(1959 -),男,浙江宁波人,教授,博士,主要研究方向:复杂非线性控制、智能控制。

种资源在任一时刻 t 的总量为 $R_t^k(k=1,2,\cdots,K;t=0,1,\cdots,T)$,$r_{jk}$ 表示任务 j 所需的资源 k 的数量,在任何时刻,项目使用的各种资源不能超过资源的总量;ST_j 为任务 j 开始执行的时间,FT_j 为任务 j 完工的时间。RCPSP 的目标就是使项目在资源均衡使用的前提下工期最短。

引入两个决策变量:

$$x_{jt}=\begin{cases}1, & \text{任务 } j \text{ 在时刻 } t \text{ 正在执行}\\ 0, & \text{其他}\end{cases} \tag{1}$$

RCPSP 可以描述为:

$$\min FT_D \tag{2}$$

$$\text{s.t.}\quad FT_j-FT_p\geqslant d_j;\forall p\in P_j,j=0,1,2,\cdots,D \tag{3}$$

$$\sum_{j=0}^{D}x_{jt}r_{jk}\leqslant R_t^k;\ k=1,2,\cdots,K;\ t=0,1,2,\cdots,T \tag{4}$$

其中:式(2)表示目标函数为项目总工期最小化;式(3)表示任务紧前关系;式(4)表示资源约束。

2　算法描述

2.1　粒子群优化算法

粒子群优化(PSO)算法是在 1995 年由 Kennedy 等[7]提出的一种具有全局寻优能力的群体智能算法。算法将种群中的每个粒子视为解空间中的一个潜在解,并通过群体的信息共享机制,使得粒子在进化寻优的过程中不断追随解空间中的最优粒子更新自身的状态。假设在一个 D 维的目标搜索空间内,粒子群由 N 个代表问题潜在解的粒子组成。粒子的状态由速度和位置向量综合表示。$\boldsymbol{x}_i=(x_{i1},x_{i2},\cdots,x_{iD})$ 表示第 i 个粒子在 D 维搜索空间内的位置,$\boldsymbol{v}_i=(v_{i1},v_{i2},\cdots,v_{iD})$ 则表示粒子 i 的飞行速度。粒子的优劣通过适应度函数来进行评价,适应度函数是粒子进行更新的驱动力。粒子 i 迄今搜索到的最优位置为 $\boldsymbol{pb}_i=(pb_{i_1},pb_{i_2},\cdots,pb_{i_D})$,整个粒子群搜索到的最优位置为 $\boldsymbol{gb}=(gb_1,gb_2,\cdots,gb_D)$。

针对目标函数 $f(x)$,假设粒子群算法求解的目标为获取函数的最小值,则粒子自身的最优位置通过式(5)进行更新:

$$\boldsymbol{pb}_i(t+1)=\begin{cases}\boldsymbol{x}_i(t+1), & f(\boldsymbol{x}_i(t+1))<f(\boldsymbol{pb}_i(t))\\ \boldsymbol{pb}_i(t), & f(\boldsymbol{x}_i(t+1))\geqslant f(\boldsymbol{pb}_i(t))\end{cases} \tag{5}$$

算法迭代寻优过程中,粒子的状态由式(6)进行更新:

$$\begin{cases}\boldsymbol{v}_i(t+1)=w\boldsymbol{v}_i(t)+c_1r_1(\boldsymbol{pb}_i-\boldsymbol{x}_i(t))+\\ \qquad\qquad c_2r_2(\boldsymbol{gb}-\boldsymbol{x}_i(t))\\ \boldsymbol{x}_i(t+1)=\boldsymbol{x}_i(t)+\boldsymbol{v}_i(t+1)\end{cases} \tag{6}$$

其中:$i=1,2,\cdots,N$;t 为当前的迭代次数;w 为惯性权重因子,用来平衡算法的全局搜索和局部搜索能力;c_1 与 c_2 分别为粒子自身学习因子和社会学习因子,它们都是非负常数;r_1 与 r_2 是区间[0,1]内的随机数。

2.2　混沌优化

将确定的运动方程得到的具有随机性的运动状态称为混沌。一个典型的混沌系统式 Logistic 映射是最常用的混沌序列发生器[8],即:

$$x_{i+1}=ux_i(1-x_i);0\leqslant x_0\leqslant 1 \tag{7}$$

其中:u 为控制参数,$x_i(i=0,1,2,\cdots,N)$ 为混沌变量。

混沌优化的步骤是首先通过序列发生器产生一组混沌变量,接着将混沌变量转换为决策变量,在转换过程中将混沌引入决策变量使其在搜索空间内呈现混沌状态,最后采用混沌化后的决策变量进行搜索。由于混沌具有遍历性,能够不重复地历经一定范围内的所有状态,比盲目无序的随机搜索更具有优越性,将其和智能算法结合可使得算法局部搜索更加有效,从而提升求解质量。

2.3　协同震荡搜索混沌粒子群算法

2.3.1　粒子与吸引子距离的确定

Clerc 通过代数和数学分析方法,对 PSO 算法中粒子收敛行为进行了分析[9],研究表明,粒子 i 的收敛过程是以点 $\boldsymbol{p}_i=(p_{i1},p_{i2},\cdots,p_{iD})$ 为吸引子更新搜索的,其坐标为:

$$p_{i,j}(t)=\frac{c_1r_{1,j}(t)pb_{i,j}(t)+c_2r_{2,j}(t)gb_j(t)}{c_1r_{1,j}(t)+c_2r_{2,j}(t)} \tag{8}$$

算法在迭代更新过程中,粒子的移动速度会不断减小,并在后期逐渐逼近 $\boldsymbol{p}_i$ 点达到收敛的状态。粒子在迭代寻优的过程中,受到 $\boldsymbol{p}_i$ 点的某种吸引力的影响,使得种群具有聚集性。

实际上,当 $c_1=c_2$ 时,式(8)可以转化成式(9):

$$\begin{cases}p_{i,j}(t)=\varphi_{i,j}(t)pb_{i,j}(t)+[1-\varphi_{i,j}(t)]gb_j(t)\\ \varphi_{i,j}(t)\sim \mathrm{U}(0,1)\end{cases} \tag{9}$$

在确立了粒子群中每个粒子的吸引子的基础上,计算出每个粒子当前所处位置和该粒子吸引子的距离 $\Delta_{i,j}(t)$:

$$\Delta_{i,j}(t)=|x_{i,j}(t)-p_{i,j}(t)|;\ 1\leqslant i\leqslant N,1\leqslant j\leqslant D \tag{10}$$

$\Delta_{i,j}(t)$ 的大小是粒子群收敛程度的反映。$\Delta_{i,j}(t)$ 的值越小代表着在吸引子的作用下粒子越靠近吸引子,种群的聚集性就越强;相反,$\Delta_{i,j}(t)$ 的值越大代表着粒子还没有及时靠近吸引子,种群相对比较发散,种群的聚集性相对较弱。在算法初期,$\Delta_{i,j}(t)$ 的值相对比较大,在后期由于迭代种群趋向同一化,$\Delta_{i,j}(t)$ 的值就会变小,粒子 i 会不断靠近吸引子 $\boldsymbol{p}_i$ 点,最终收敛到 $\boldsymbol{p}_i$ 点[10]。文献[10]的 RSPSO 算法采用式(10)进行震荡搜索,而本文采用式(10)驱动粒子向吸引子靠近,具体实现过程在 2.3.3 节中介绍。

2.3.2　粒子震荡搜索范围的确定

定义 1　邻距向量。邻距向量表示单个粒子中后一维度和前一维度的差值。邻距向量中第一个元素保持和粒子中的一致。如粒子 $\boldsymbol{x}_i=(0.12,0.45,0.55,0.86,0.24,0.39)$,则它的邻距向量 $\boldsymbol{L}_i=(0.12,0.33,0.10,0.31,-0.62,0.15)$。

按照邻距向量的定义,分别计算出吸引子 $\boldsymbol{p}_i$ 的邻距向量 $\boldsymbol{U}_i$ 和 $\boldsymbol{x}_i$ 的邻距向量 $\boldsymbol{L}_i$,然后判断邻距向量中各维度的正负,邻距向量各维度的正负反映前后维度的大小关系,这将决定粒子中哪些维度需要震荡搜索。$\boldsymbol{U}_i$ 和 $\boldsymbol{L}_i$ 各维的正负符号通过式(11)来计算:

$$\begin{cases}\boldsymbol{flag1}=\mathrm{sign}(\boldsymbol{U}_i), & i=1,2,\cdots,N\\ \boldsymbol{flag2}=\mathrm{sign}(\boldsymbol{L}_i), & i=1,2,\cdots,N\end{cases} \tag{11}$$

其中:$\boldsymbol{flag1}$ 反映吸引子前后维度之间的大小关系,$\boldsymbol{flag2}$ 则反映粒子前后维度的大小关系。由于粒子在收敛过程中向吸引子靠近,所以本文采用粒子效仿吸引子前后维度的关系大小来调整自身的前后维度的关系大小。粒子需要调整的维度是和吸引子前后维度关系不一致的维度。查找粒子和吸引子不一致的维度可以根据式(12)查找 $\boldsymbol{M}_i$ 中维度值为 -1 的集合。

$$M_{i,j}=flag1_j*flag2_j;\ i=1,2,\cdots,N,\ j=1,2,\cdots,D \tag{12}$$

Journal of 《计算机应用》 Computer Applications

粒子前后维度关系震荡搜索的方向由 ***flag*1** 确定，$flag1_j$ 若为 1，则 x_{ij} 需向上搜索，即值变大；反之，若值为 -1，则 x_{ij} 需向下搜索，即值变小。

需要震荡搜索的粒子维度的搜索范围需要吸引子邻距向量和粒子的邻距向量共同界定。这里的粒子震荡搜索范围只是对于需要改变粒子前后维度大小关系的维度集合来讲的，不需要改变的维度集合保持原来的状态。假定吸引子的邻距向量 $\boldsymbol{U}_i$ 中需要改变维度关系的子向量为 $\boldsymbol{U}_i'$，粒子邻距向量 $\boldsymbol{L}_i$ 中需要改变维度关系的子向量为 $\boldsymbol{L}_i'$，则需调整相应维度的最大幅度 $\Delta_{i,j}'(t)$ 为：

$$\Delta_{i,j}'(t) = |U_{i,j}'(t)| + |L_{i,j}'(t)|;$$
$$i = 1,2,\cdots,N, j = 1,2,\cdots,D \tag{13}$$

$\Delta_{i,j}'(t)$ 用来限制需要震荡搜索的粒子维度移动的最大步长，则粒子需震荡搜索维度的范围为 $[0,\Delta_{i,j}'(t)]$，通过在搜索范围内区域震荡调整需要变化位置的粒子维度的值，来寻找到比自身历史极值更佳的值。粒子搜索的过程中，区域震荡因子 $\Delta_{i,j}'(t)$ 线性减小，区域震荡因子线性减少的公式为：

$$\Delta_{i,j}'(k) = \Delta_{i,j}'(t) - \Delta_{i,j}'(t) * k/RS \tag{14}$$

RS 为粒子最大的震荡次数，*k* 则为当前的粒子震荡次数。

2.3.3 协同震荡搜索策略

为了方便和快速地使粒子向吸引子靠拢，提出双向协同震荡搜索的方法来寻找最优解，一方面粒子向吸引子收敛，另一方面粒子效仿吸引子前后维度的关系大小来震荡调整自身的前后维度的关系大小，在保证粒子收敛的同时，提升了粒子的多样性。

粒子各维度向吸引子靠拢的位置更新式为：

$$x_{i,j}(t+1) = x_{i,j}(t) + flag * w * \Delta_{i,j}(t);$$
$$flag = \begin{cases} 1, & (x_{i,j}(t) - p_{i,j}(t)) \leqslant 0 \\ -1, & (x_{i,j}(t) - p_{i,j}(t)) > 0 \end{cases} \tag{15}$$

其中：$x_{i,j}(t)$ 为粒子当前所处的位置；*flag* 用来决定粒子搜索的方向，使粒子在每次搜索的过程中都保证粒子是向着吸引子靠近；*w* 为调节系数，类似式(6)中的惯性权重 *w*，用来自适应动态调整粒子移动的步长，*w* 的值的变化影响粒子收敛的速度和精度。

协同搜索除了向粒子吸引子收敛外，加入了改变粒子前后维度关系与吸引子不一致的搜索策略，使得粒子双向搜索。粒子各维度效仿吸引子前后维度关系震荡调整粒子位置的更新式：

$$x_{i,j}'(k) = x_{i,j}'(t) + flag1_j * \Delta_{i,j}'(k);$$
$$k = 1,2,\cdots,RS, j = 1,2,\cdots,D \tag{16}$$

其中：$x_{i,j}'(t)$ 为 *t* 次迭代过程中，需要调整粒子前后维度大小关系的粒子维度；$x_{i,j}'(k)$ 为该维度第 *k* 次震荡更新时的位置。粒子位置结合式(15)和(16)进行更新，粒子的每个维度都要按照(15)来进行更新，在完成式(15)的更新的基础上，需要调整粒子前后维度大小关系的维度集合再按照式(16)来进行震荡更新搜索，当完成设定的粒子区域震荡次数 *RS* 后，根据适应度值选取震荡搜索中获取到的最优粒子，并结合粒子的状态更新粒子历史极值和种群全局极值。由于震荡搜索的策略的加入，使得粒子在收敛过程中，增加了种群多样性，在求解项目调度问题中提升在搜索空间内获得最优解的概率。

3 CSCPSO 算法求解 RCPSP

3.1 粒子表示与初始化

本文采用粒子位置结合拓扑排序生成项目调度次序，粒子各维度的大小将决定拓扑序列中的次序。以粒子群的搜索空间维数来代表项目的任务数，总数为 *D*，粒子采用混沌变量初始化，变量值的大小决定选择没有先序任务或其先序任务都已经分配资源的任务集合中任务的优先权。在 RCPSP 模型中，应用拓扑排序结合粒子每维的值生成项目的调度次序，保证了生成的调度方案满足紧前约束。

以图 1 为例，图中任务 1 和 6 是虚任务，代表开始和结束任务，在实施过程中不消耗任何时间和资源。随机生成一个粒子 $\boldsymbol{x}_i$ = (0.583 2, 0.683 7, 0.901 2, 0.703 2, 0.501 1, 0.453 6)，按照拓扑排序的方法，首先确定任务 1，因为任务 1 没有紧前约束。接着在任务 2 和任务 3 中选择，任务 2 和 3 在拓扑排序中的地位是同等的，但是把哪一个排在前面，取决于对应粒子维度的值，任务 2 对应粒子维度的值为 0.683 7，而任务 3 对应的粒子维度的值为 0.901 2，本文采用维度值大的优先排列，则任务 2 和任务 3 优先选择任务 3。任务 3 确定后，任务 2 和任务 5 的地位是同等的，同样采取维度值大者优先的规则来选择，最后产生一个调度方案 1-3-2-4-5-6，这样生成的调度次序不用调整粒子位置就保证满足紧前约束条件。然后按照 3.2 节的调度生成方式计算适应度值。

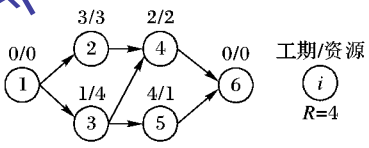

图 1 项目网络图

3.2 调度生成方式

本文采用串行调度方案求解，项目的调度顺序由粒子的位置决定，根据生成的满足紧前约束条件的调度次序，依次执行。其中 ST_j 为任务 *j* 的最早开始时间，取其所有紧前任务的最早结束时间中的最大值。调度顺序中如果任务 *j* 与不是它的先序任务的 *j* - 1 并行执行满足资源约束，则选择并行操作；否则选择串行操作。通过串行调度方案可以将 3.1 节中的粒子 $\boldsymbol{x}_i$ 转化成如图 2 所示的工期为 6 的调度。

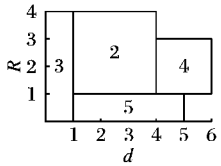

图 2 项目调度图

3.3 算法流程

步骤 1 混沌初始化种群及相关参数。种群规模 *N*，每个粒子的维度 *D*，种群的迭代次数 *MaxDT*，粒子震荡搜索次数 *RS*；种群的初始位置采用式(7)混沌序列发生器产生，并确定每个粒子的个体最优位置和种群的全局最优位置。

步骤 2 计算出每个粒子吸引子 $\boldsymbol{p}_i$ 的位置。

步骤 3 根据 $\boldsymbol{p}_i$ 的位置结合拓扑排序算法生成对应的项目调度次序。

步骤 4 根据吸引子生成的任务调度次序分别计算出 $\boldsymbol{p}_i$

《计算机应用》Journal of Computer Applications

按照调度顺序安排的各维邻距向量 $\boldsymbol{U}_i$,以及粒子 $\boldsymbol{x}_i$ 参照吸引子的调度顺序各维邻距向量 $\boldsymbol{L}_i$。

步骤 5　协同震荡搜索调整粒子位置。

步骤 5.1　根据式(10)计算出粒子每维与吸引子间的距离 $\Delta_{i,j}(t)$;

步骤 5.2　根据式(13)确定需要震荡搜索维度集合的震荡搜索范围;

步骤 5.3　结合式(15)与(16)进行双向协同震荡搜索,每震荡搜索一次产生一个调度方案,并按照 3.2 节的调度生成方式计算工期。

步骤 6　根据适应度值,选取步骤 5 中搜索到的最优粒子位置,更新粒子历史最优和全局最优位置。

重复步骤 2 至步骤 6,直到满足迭代的次数。

4　算例

为了验证 CSCPSO 算法求解 RCPSP 的有效性,本文采用了文献[11]中的实际案例,其网络结构如图 3 所示,来进行检验并和该文献中用到的算法获取的结果进行比较。

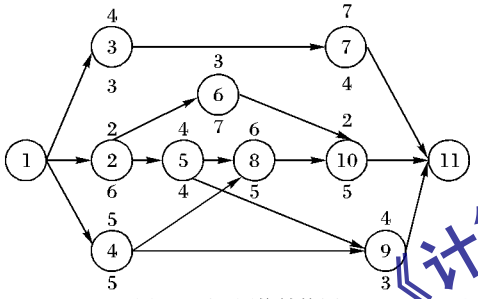

图 3　项目网络结构图

图 3 中任务节点上方为任务需要完工所需时间,节点下方为该任务所需的资源,任务资源约束条件 R = 12,运用 CSCPSO 算法对图 3 描述的项目进行求解,设定 $MaxDT$ = 20,w 随着迭代次数线性减少,初始值为 1.6,结束值为 0.4,粒子规模 N = 20,粒子震荡搜索次数 RS = 10。为求准确,算法独立运行 100 次来检验结果,算法运行结果如表 1 所示,并采用如下指标来评价:算法所得最优解,100 次测试所得解与所获最优解差距的最大值、最小值以及达到最优解次数的百分比。

从表 1 可看出:CSCPSO 算法在求解项目调度工期最小值上,相对于文献[11]算法,本文算法将项目工期缩短了 1 天,并将获取最优值的成功率提升了 1 倍;RSPSO 算法虽然能够求得项目的最优值,但是在求解的稳定性上略显不足,这是因为粒子在寻优时维度之间缺乏信息沟通,采用例子表示任务的优先权值时,项目在调度过程中无法灵活地改变任务的优先权,使得在求解质量上表现欠佳;而 CSCPSO 算法对项目进行优化求解时灵活改变任务的优先权每次都可以获取到最优值,并且与最优值之间没有偏差,表现出良好的稳定性与寻优能力。

表 1　项目调度结果比较

算法	获取最优值	最小差值	最大差值	获取最优值的百分比/%
RSPSO[10]算法	16	0	1	80
文献[11]算法	17	0	5	51
CSCPSO 算法	16	0	0	100

图 3 中的实例任务数偏少,而且任务所需资源只有一种,属于比较简单的调度问题。本文继续以国际标准问题库 http://129.187.106.231/psplib 中 J301-1.SM 为测试问题,并用 CSCPSO 算法求解最小工期。作业信息如表 2 所示。

CSCPSO 算法对表 2 描述的项目进行求解,设定 $MaxDT$ = 50,粒子规模 N = 20,震荡次数 RS = 10。算法独立运行 1 000 次,与文献[12]中对于同一项目的调度结果进行比较,结果如表 3 所示。

从表 3 可看出:对于 J301-1.SM 项目的求解,CSCPSO 较文献[12]中的两个算法有着更加优秀的寻优效果,每次均能找到项目的最短工期,并且在最优方案的平均偏差上体现出较强的稳定性,有效解决复杂项目问题调度问题。

表 2　J301-1.SM 作业信息表

作业号	作业时间	紧后作业号	所需资源数			
			A	B	C	D
1	0	2,3,4	0	0	0	0
2	8	6,11,15	4	0	0	0
3	4	7,8,13	10	0	0	0
4	6	5,9,10	0	0	0	3
5	3	20	3	0	0	0
6	8	30	0	0	0	8
7	5	27	4	0	0	0
8	9	12,19,27	0	1	0	0
9	2	14	6	0	0	0
10	7	16,25	0	0	0	1
11	9	20,26	0	5	0	0
12	2	14	0	7	0	0
13	6	17,18	4	0	0	0
14	3	17	0	8	0	0
15	9	25	3	0	0	0
16	10	21,22	0	0	0	5
17	6	22	0	0	0	8
18	5	20,22	0	0	0	7
19	3	24,29	0	1	0	0
20	7	23,25	0	10	0	0
21	2	28	0	0	0	0
22	7	23	2	0	0	0
23	2	24	3	0	0	0
24	3	30	0	9	0	0
25	3	30	4	0	0	0
26	7	31	0	0	4	0
27	8	28	0	0	0	7
28	3	31	0	8	0	0
29	7	32	0	7	0	0
30	2	32	0	7	0	0
31	2	32	0	0	2	0
32	0	—	0	0	0	0

注:可更新资源 A、B、C、D 的总量分别为 12,13,4,12。

表 3　J301-1.SM 项目调度结果比较

算法	获取最优值	距最优方案平均偏差/%	达到最优值次数的百分比/%
BPSO[12]算法	43	0.49	98.9
CPSO[12]算法	43	0.30	99.6
CSCPSO 算法	43	0.00	100.0

以上两个实例都是针对单项目的调度,资源受限多项目

Journal of Computer Applications 《计算机应用》

调度相对于单项目调度来说,项目数和任务数都更多,资源冲突会更加明显,调度过程因此会更加复杂。本文将图 4 中的项目 1 和图 5 中的项目 2 进行组合构成一个多项目来对算法进行检验,两个项目共用 R_1 和 R_2 两种可再生资源,总量分别为 10 和 12,两个独立项目的网络结构如图 4 ~ 5 所示,节点上面的数字为该任务的工期,节点下面的数字为两者资源的消耗量。

对于项目 1 和项目 2 结合的多项目的求解,为避免算法的随机性,算例进行 100 次计算,参数设置与文献[13]中的一致,算法所得解的分布如图 6 所示,并将计算的结果同文献[13]进行比较,项目调度结果比较如表 4 所示。

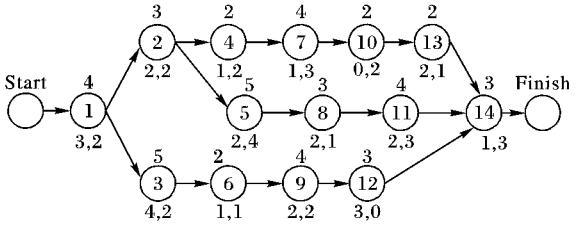

图 4 项目 1 网络结构图

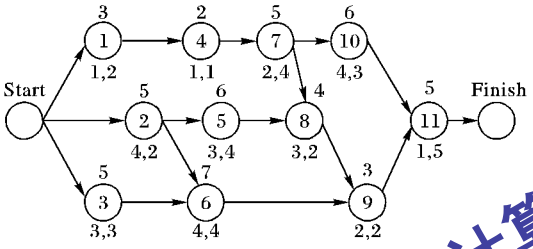

图 5 项目 2 网络结构图

表 4 多项目调度结果比较

算法	达到最优解次数	平均工期
SHO[13]算法	92	30.23
PSO[13]算法	84	32.06
CSCPSO 算法	见图 6	29.23

由图 6 的结果分布图得出,本文算法在求解多项目调度过程中,获得的最差工期为 30,结合表 4 的信息可以得出,CSCPSO 算法取得的最差值已然等于或优于文献[13]两个算法所取得的最优值。100 次独立测试下,CSCPSO 算法找到了更加优秀的调度方案,并将平均工期缩短了一天,在多项目问题调度中同样体现出良好的寻优性能。

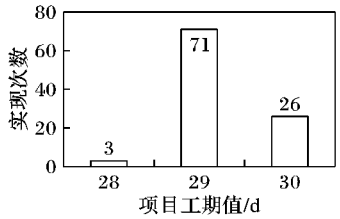

图 6 项目工期结果分布图

5 结语

本文针对求解资源受限项目调度问题(RCPSP),提出了协同震荡搜索混沌粒子群(CSCPSO)算法。实验结果表明,CSCPSO 在求解满足约束条件下的最短工期问题上具有较高的准确率,克服了传统 PSO 易陷入局部最优解的缺点,且拥有良好的稳定性,验证了算法的有效性。CSCPSO 相比传统 PSO 没有速度向量,实现更为简单,进一步提升了 PSO 算法在项目调度领域的应用价值。不过,在实际情况中,出于人性化方面的考虑,还应考虑到资源使用间隔、影响资源疲劳度的各种因素等,这些可在以后进一步研究。

参考文献:

[1] VALLS V, BALLESTIN F, QUINTANILLA S. Justification and RCPSP: a technique that pays [J]. European Journal of Operational Research, 2005, 165(2): 375 - 386.

[2] KOLISCH R, HARTMANN S. Experimental investigation of heuristics for resource-constrained project scheduling: an update [J]. European Journal of Operational Research, 2006, 174(1): 23 - 37.

[3] WANG H, LIN D, LI M. Genetic algorithm for multi-objective resource-constrained project scheduling problem [J]. Computer Engineering and Applications, 2008, 44(7): 1 - 4. (王宏, 林丹, 李敏强. 一种求解多目标资源受限项目调度的遗传算法[J]. 计算机工程与应用, 2008, 44(7): 1 - 4.)

[4] MERKLE D, MIDDENDORF M, SCHMECK H. Ant colony optimization for resource-constrained project scheduling [J]. IEEE Transactions on Evolutionary Computation, 2002, 6(4): 333 - 346.

[5] YU X, ZHAN D, NIE L. Genetic simulated annealing algorithm for resource-constrained project scheduling problem [J]. Computer Engineering and Applications, 2009, 45(24): 17 - 20. (喻小光, 战德臣, 聂兰顺. 应用遗传模拟退火算法实现资源受限项目调度[J]. 计算机工程与应用, 2009, 45(24): 17 - 20.)

[6] WANG W, ZHAO G. Particle swarm optimization for resource-constrained project scheduling problem[J]. Journal of Harbin Institute of Technology, 2007, 39(4): 669 - 672. (王巍, 赵国杰. 粒子群优化在资源受限工程调度问题中的应用[J]. 哈尔滨工业大学学报, 2007, 39(4): 669 - 672.)

[7] KENNEDY J, EBERHART R C. Particle swarm optimization [C]// Proceedings of the 1995 IEEE International Conference on Neural Networks. Piscataway: IEEE Press, 1995: 1942 - 1948.

[8] CAPONETTO R, FORTUNA L, FAZZINO S. Sequences to improve the performance of evolutionary algorithms [J]. IEEE Transactions on Evolutionary Computation, 2003, 7(3): 289 - 304.

[9] CLERC M, KENNEDY J. The particle swarm: explosion, stability, and convergence in a multi-dimensional complex space [J]. IEEE Transactions on Evolutionary Computation, 2002, 6(1): 58 - 73.

[10] TANG J, DAI Y. Regional shock search embedded particle swarm optimization algorithm [J]. Computer Engineering and Applications, 2013, 49(21): 33 - 36. (汤继涛, 戴月明. 内嵌区域震荡搜索的粒子群优化算法[J]. 计算机工程与应用, 2013, 49(21): 33 - 36.)

[11] ZHANG K, ZHAO G, JIANG J. Particle swarm optimization method for resource-constrained project scheduling problem [C]// ICEMI 2009: Proceedings of the Ninth International Conference on Electronic Measurement & Instruments. Piscataway: IEEE Press, 2009: 792 - 796.

[12] XIE Y, YE C, CHEN J, *et al*. Resource - constrained project scheduling based on chaos particle swarm optimization [J]. Industrial Engineering Journal, 2012, 15(3): 57 - 61. (谢阳, 叶春明, 陈君兰, 等. 基于混沌粒子群的资源受限项目调度问题[J]. 工业工程, 2012, 15(3): 57 - 61.)

[13] NI L, DUAN C, ZHONG H. Multi-project scheduling based on simulated harmonic oscillator algorithm [J]. Journal of Computer Applications, 2011, 31(9): 2559 - 2562. (倪霖, 段超, 钟辉. 基于模拟谐振子算法的多项目调度[J]. 计算机应用, 2011, 31(9): 2559 - 2562.)

《计算机应用》Journal of Computer Applications