基于 IPSO 算法的多 Agent 联盟形成研究

陈宁霞 刘大勇

(江西信息应用职业技术学院 江西南昌 330043)

摘 要:考虑到单个 Agent 资源有限,多个 Agent 往往需形成联盟来完成任务或提高联盟整体的能力,如何形成一组针对某个任务的最佳联盟是 MAS 中一个紧迫而又关键性的问题。基于此文中提出一种改进的 IPSO 算法来解决该问题,同时为克服粒子过早收敛和局部优化,在改进的惯性权重上引入一种柯西变异的扰动算子,最后与 PSO 算法及 ACO 算法做对比。结果表明该 IPSO 算法的全局搜索能力较高,成功地避免了粒子过早收敛,资源浪费等。

关键词:多 Agent 联盟;PSO 算法;ACO 算法;IPSO 算法

Research on the Formation of Multi-Agent Alliance Based on IPSO Algorithm

Chen Ningxia Liu Dayong

(Jiangxi Vocational & Technical College of Information Application 330043)

Abstract: Considering the finiteness of the single agent ability, multiple agents often need to form a coalition to accomplish a task or improve the whole ability of coalition. The primary issue in the study of MAS is how to generate an optimal coalition oriented a set of tasks. Based on this background, this paper puts forward an improved particle swarm optimization algorithm (IPSO) to solve the problem. In order to overcome the particle premature convergence and local optimization problems, a Cauchy mutation disturbance operator was introduced on the basis of improved inertial weight. Compared with PSO algorithm and the ACO algorithm, the global optimization of IPSO algorithm is better, which effectively avoided the premature convergence, the wasting resources problems and so on.

Key Words: multi-agent system; PSO algorithm; ACO algorithm; IPSO algorithm

1 Agent 联盟形成问题描述

设 MAS 中含有 n 个 Agent,即 A={$a_1,a_2,\cdots,a_n$},集合中可构成 2^n 个联盟,需合作求解 m 个任务,其中$C_i(1\le i\le 2^n,\ C_i\neq\varnothing)$表示其中的任意一个联盟。设任务集 T={$t_1,t_2,\cdots,t_n$},m 个任务间的排列顺序与任务的优先级有关,不妨设 $t_1>t_2>\cdots>t_m$,优先级高的任务被优先执行。具体设置如下:

每个 Agent $a_j\in A$ 都有 r 维的能力向量$B_j=[b_j^j,b_2^j,\ldots,b_r^j]$,$b_k^j$ 是 a_j 的第 k 维能力分量,$0\le b_k^j<\infty, j\in\{1,2,\ldots n\}, k\in\{1,2,\ldots r\}$;

每个联盟 C_i 具有的能力由向量 $B_{C_i}=(b_1^{C_i},b_2^{C_i},\ldots,b_r^{C_i})$ 表示。若联盟 C_i 完成任务 T 则获得的全部费用为$Value_T, Value_T>0$。联盟 C_i 一般具有两个属性:联盟代价 $Cost_{ci}$ 和联盟值 Profit$_{ci}$。$Cost_{ci}$ 为执行任务所付出的成本,Profit$_{ci}$ 为完成任务 T 联盟 C_i 可获得的收益。由以上可知 Profit$_{ci}$ 为 $Cost_{ci}$ 和 $Value_T$ 共同决定,且 Profit$_{ci}\ge 0$(Profit$_{ci}$=0 表此联盟不能完成该任务);

每个 Task $t_i\in T$都有 r 维的资源需求向量$D_i=[d_1^i,d_2^i,\ldots,d_r^i]$,$d_k^i$ 是 t_i 的第 k 维能力需求分量,且$0\le d_k^i<\infty, i\in\{1,2,\ldots m\}$,任务 t_i 被联盟 C_i 完成必要条件是:$b_k^{C_i}\ge d_k^i, k\in\{1,2,\ldots r\}$。为加速 Agent 联盟形成,通常将联盟限于超加性环境:即对任意两个联盟 $C_1,C_2\subseteq N$,若 $C_1\cap C_2=\varnothing$,则 $V(C_1\cup C_2)\ge V(C_1)+V(C_2)$,且联盟越大效用越多,联盟扩大所增加效用为△V,△V=$V(C_1\cup C_2)-V(C_1)-V(C_2)$。

2 几种常用求解多 Agent 联盟生成算法

作者简介:陈宁霞(1986—),女,助教,主要研究方向:软件开发、数据库设计、前端设计。

联盟形成是 MAS 研究中一个紧迫而又关键性的问题，目前对其的研究很多，现将几种常见求解多 Agent 联盟生成算法归纳如下：

2.1 GA 算法求解 Agent 联盟

遗传算法（Genetic Algorithm,GA）作为群智能算法中的一种，其主要是基于自然选择和遗传学原理，具体流程为：GA 首先随机产生一些由个体组成的初始个体群 H_i ($H_i=\{A_1,A_2,A_3,\cdots,A_n\}$)，根据目标函数决定适应度函数，计算各个个体的适应度；其次根据适应度值对个体进行选择，留下适应度高的个体，然后进行交叉、变异产生进化群体，如此反复不断向最优解进化；最后得到满足某种收敛条件的最适应个体，进而获得问题最优解。其中 $A_i=1$ 表 a_i 在联盟中，$A_i=0$ 表 a_i 不在联盟中。GA 求解多 Agent 联盟的关键步骤如下：

基因编码；

初始化：设置一个整数 p 作为群体规模，然后随机产生 p 个位数为 n 的二进制编码串作为初始种群，定义适应度函数为 $F(C)=Value_c=Profit_c/Cost_c$，其中 $Profit_{ci}=0$ 表任务不能被联盟完成；

选择：群体中每个个体都有被选择的可能；

交叉：即将两个个体进行随机交叉配对；

变异：即按照概率改变染色体中某个基因位；

替换：为找到最好解，一般将子群体中适应度最小个体用最优个体代替。

2.2 ACO 算法求解 Agent 联盟

蚁群算法（Ant Colony Algorithm,ACO）是一种新型模拟进化算法，其主要模拟真实蚁群合作过程，并在所寻解上留下一定信息量，信息量越大表明解的性能越好。算法初期所有解上的信息量是相同的，随着算法推进较优解上的信息量也增加直至算法收敛。针对单任务 ACO 算法求解联盟时模型如下：

设 m 是蚂蚁的数量，$d_{ij}(i,j=1,2,\cdots,n)$表示 a_i 和 a_j 之间的距离，即通信开销。p^k_{ij} 表在 t 时刻蚂蚁 k 由位置 i 转移到位置 j 的概率，即 t 时刻蚂蚁 k 选择 a_j 加入联盟的概率，其表达式如(3–1)示。$\tau_{ij}(t)$表 t 时刻在 ij 连线上残留的信息素量，表“熟悉度”或“好友度”，其更新式如(3–2)、(3–3)所示，其中 ρ 表示熟悉度的遗忘，$\Delta\tau^k_{ij}$表第 k 只蚂蚁在本次循环中对 a_i,a_j 之间熟悉度的增量，$V(C_k)$是蚂蚁 k 形成的联盟值。

$$p^k_{ij}=\begin{cases}\dfrac{[(\tau_{ij}t)]^\alpha[1/d_{ij}]^\beta]}{\sum[(\tau_{iu}t)]^\alpha[1/d_{iu}]^\beta} & j\in J_k\\ 0, \quad \text{其他}\end{cases} \tag{3–1}$$

$$\tau_{ij}(t+1)\leftarrow\rho\cdot\tau_{ij}(t)+\sum_{k=1}^{m}\Delta\tau^k_{ij} \tag{3–2}$$

$$\Delta\tau^k_{ij}=\begin{cases}\dfrac{V(C_k)}{\sum_{k=1}^{m}V(C_k)}, & \text{蚂蚁}k\text{形成联盟包含}a_i,a_j\\ 0 & \text{其它}\end{cases} \tag{3–3}$$

针对多任务 ACO 算法求解联盟时模型如下：

首先根据任务紧迫度 $U(t_j)$对 T 中任进行序；其次寻找能够完成任务 t_j 且联盟值最大的联盟 C，当求出任务 t_j 的最优联盟后再求下一任务 t_{j+1} 的最优联盟，此时 Agent 间的信息素不再是初始值τ_0（为便于求出信息素文献引入一个内激素 $R=0.9^g$，具体如式(3–4)所示），而是在上一次求解结束时残留下来的信息素$\tau_{ij}(t)$即式(3–3)所求得的信息素，以此类推直到所有任务都被顺利完成。

$$g=\begin{cases}0, & \text{算法求得最优解仍在进化}\\ g+1, & \text{最优解在}N\text{次迭代后无明显改进，且}g+1\le g_{\max}\\ g_{\max} & \text{其它}\end{cases} \tag{3–4}$$

2.3 PSO 算法求解 Agent 联盟

对 n 个 Agent 集合 $N=\{A_1,A_2,\cdots,A_n\}$，用 PSO 算法求解 Agent 联盟时，先假设 PSO 算法处在一个 D 维空间中，每个粒子的位置可表示为 $x^t_i=(x^t_{i1},x^t_{i2},\ldots,x^t_{iD})$，具体如式(3–7)；对应粒子速度可表示为 $v^t_i=(v^t_{i1},v^t_{i2},\ldots,v^t_{iD})$，其中$v^t_{id}\in[v_{\min},v_{\max}]$，$V_{\min},V_{\max}$ 分别为粒子速度更新的最小最大区域边界。以二维空间为例图 3–1 列了粒子按照式(3–5)和(3–6)的模型移动。

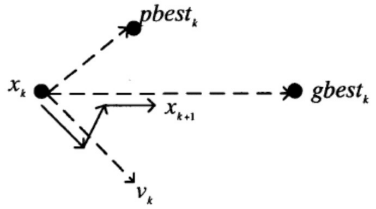

图 3–1 粒子移动矢量示意图

算法中具体粒子速度的更新公式为：

$$v_{id}(t+1)=v_{id}(t)+r_1*c_1*(p_{id}(t)-x_{id}(t))+r_2*c_2*(p_{gd}(t)-x_{gd}(t)) \tag{3–5}$$

$$v_{id}=\begin{cases}v_{\max}, if & v_{id}>v_{\max}\\ -v_{\max}, if & v_{id}<-v_{\max}\end{cases} \tag{3–6}$$

粒子调整自身位置更新公式为：

$$x_{id}(t+1)=v_{id}(t)+x_{id}(t) \tag{3–7}$$

为更好控制寻优范围且降低 V_{max} 对寻优结果的影程度，Shi 和 Eberchart 博士又引入惯性权重对式(3–5)进行更新，结果如式(3–8)所示：

$$v_{id}(t+1)=\omega*v_{id}(t)+r_1*c_1*(p_{id}(t)-x_{id}(t))+r_2*c_2*(p_{gd}(t)-x_{gd}(t)) \tag{3–8}$$

其中当 ω=1 即为 PSO 法，一般 ω 式(3–9)调整：

$$\omega=\omega_{\max}-(\omega_{\max}-\omega_{\min})\frac{t}{T_{\max}} \tag{3–9}$$

T_{max} 为最终迭代次数，ω_{max} 为初始值通常取 0.9，ω_{min} 为最小值通常取 0.4。

3 基于改进的 PSO 算法求解多 Agent 联盟

3.1 基于惯性权重 w 的改进

基于文献可知，当 w 很大时将会帮助扩大搜索范围，当 w 很小时则有利于小范围内搜索。为促进粒子更好地搜索，对 w 的值进行改进如式(3–10)所示。

$$w_{t+1} = w_t - w_t * (\frac{t+1}{T_{\max}}) \tag{3-10}$$

w_{t+1} 为 w 第 t+1 次的，经验证该方法可提升收敛速度，但考虑到 PSO 易陷入早熟难以避免粒子陷入局部最优，因此对粒子位置需做进一步改进。

3.2 自适应柯西变异算子的引入

3.2.1 柯西分布原理

柯西分布是数学中一类常见分布，具体公式为：

$$f_t(x)=\frac{1}{\pi}\frac{t}{t^2+x^2}\ ,\ -\infty<x<\infty \tag{3-11}$$

其中，t 为比例参数，且 $t\succ 0$ ，相应函数定义为：

$$F_t(x)=\frac{1}{2}+\frac{1}{\pi}\arctan\left(\frac{x}{t}\right) \tag{3-12}$$

其中，当 t=1 时，称为标准柯西分布，如图 3-2 所示：

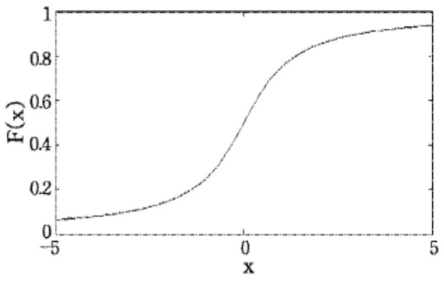

图 3-2 标准柯西分布函数

3.2.2 柯西变异算子

设 ξ 是服从均匀分布的随机变量且 $\xi\in[0,1]$，由文献定理可知，Cauchy(0,1)柯西分布的随机变量生成函数为：

$$f(x)=\tan[(\xi-0.5)\pi] \tag{3-12}$$

文中柯变异算子思想是：对于种群中的个体，个体根据式(3-13)进行自适应变异，并将求得的结果取代个体中最差粒子的位置：

$$X_i^{'}=x_i+x_i*Cauchy(0,1) \tag{3-13}$$

$$t=\frac{P_{id}}{P_{gd}} \tag{3-14}$$

其中，Cauchy(0,1)为标准柯西分布，t 表示粒子当前个体最优值和当前全局最优值的比值。因考虑粒子的局部最优和全局最优位置，经验证粒子收敛性得到很大提高。

3.3 粒子适应值的计算

如果该粒子当前的适应值等于个体历史最优值或粒子当前的适应值等于全局粒子最优值，则随机选择粒子位置；如果该粒子当前的适应值优于个体历史最优值或粒子当前的适应值优于全局粒子最优，则将历史最优值或全局最优值替换当前粒子的位置；

3.4 算法步骤

Step1 初始化种群，设粒数为 n、学习因子 c_1,c_2 及惯性权重 w 初始值、最大迭代次数的值，并随产生 n 个初位置及初始速度；

Step2 根据适应度函数对每个粒子的位置进行更新；

Step3 根据式(3-10)对 w 的值进行更新，同时由式(3-7)和(3-8)计算下一刻粒子的位置和速度，并运式(3-13)和(3-14)对粒子的位置进行扰动，同时记下每个粒子的当前值；

Step4 若粒子的当前值优于个体历史最优值，则将当前个体最优值替换个体历史最优值，否则不替换；

Step5 若粒子的当前值优于群体历史最优值，则将当前个体最优值替换群体历史最优值，否则不替换；

Step6 若达到最高迭代次数或找到最优解，则输出搜索结果，否则转入 Step2；

4 实验环境设置及结果分析

为验证算法性能，文中以 ACO、PSO 及本文的 IPSO 三种算法进行实验设计。假定任务量分别为 4、12 和 30，具体算法参数假设下：粒子数 n=30，c_1=c_2=2，w 初始值为 0.9，Agent 的数目为 200，最大迭代次数为 1000。每个 Agent 的能力向量 $A_i=<b_i^1,b_i^2,b_i^3,b_i^4>$ ，A_i 完成任务需要的联盟代价值为 $Cost_{A_i}=1\times a_i^1+2\times a_i^2+3\times a_i^3+4\times a_i^4$；任务需求能力向量 $B_l=<b_l^1,b_l^2,b_l^3,b_l^4>$ ，完成任务需要的联盟值为 $\Pr ofit_l=1\times b_l^1+2\times b_l^2+3\times b_l^3+4\times b_l^4$。图 3-3 和图 3-4 列出了 IPSO 算法的收敛曲线及粒子群体多样性的变化曲线，可看出 IPSO 算法收敛性较好及粒子的群体多样性始终保持较高水平，很好地防止了粒子陷入局部最优值。

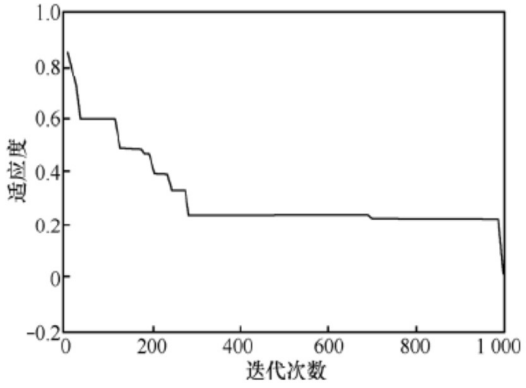

图 3-3 粒子收敛过程

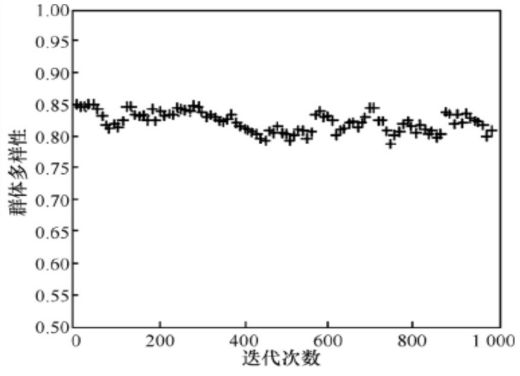

图 3-4 粒子群体多样性变化

4.1 实验环境设置

当 $B_{all}=\sum_{i=1}^{n}B_{A_i}<B_l$ 时，因 Agent 能力不足，联盟失败，故任务均不能被完成；

当 $B_{all}=\sum_{i=1}^{n}B_{A_i}<B_l$ 时，符合联盟生成条件，由表 3-1 及图 3-5、图 3-6 可知，IPSO 算法的联盟值优于 PSO 和 ACO 算法，且迭代次数越大，联盟值的差距越大。

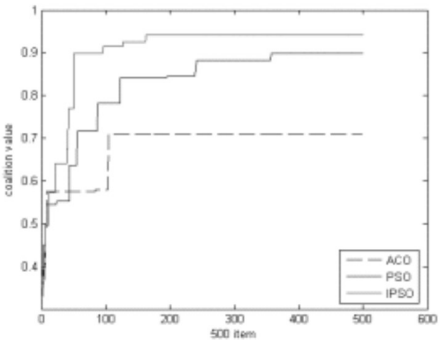

图 3-5 最优解的变化曲线(迭代 500 次)

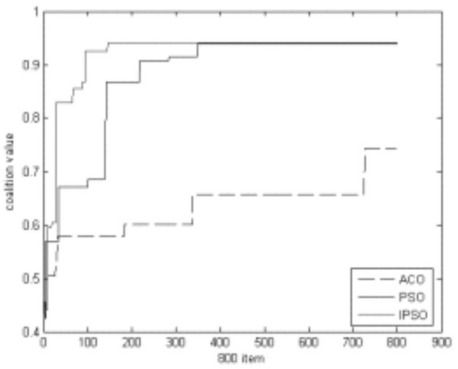

图 3-6 最优解的变化曲线(迭代 800 次)

表 3-1 迭代次数为 500 三种算法下联盟值大小比较

实验环境	任务数	联盟最优值		
		IPSO	PSO	ACO
1	4	0	0	0
	12	0	0	0
	30	0	0	0
2	4	0.9372	0.9361	0.9012
	12	0.9313	0.9282	0.8819
	30	0.9295	0.9081	0.7854
3	4	0.8479	0.8312	0.8237
	12	0.7164	0.6961	0.6772
	30	0.5368	0.4876	0.4812

当 $B_{all}=\sum_{i=1}^{n}B_{A_i}\succ\succ B_l$ 时，由表 3-2 及图 3-7、图 3-8 可知与 ACO 和 PSO 算法相比，IPSO 算法的联盟值还是最高的，但存在资源浪费，并随迭代次数增加，联盟值的差距不是很大，这是因为当 Agent 能力远远高于任务能力需求时，只要 Agent 能完成任务，就可获得一个很好的收益，最优联盟就显得不是那么重要了。

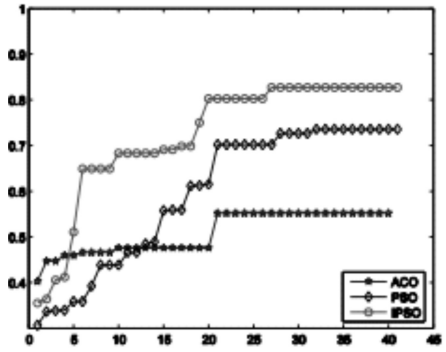

图 3-7 三种算法下最优解的变化曲线(500)

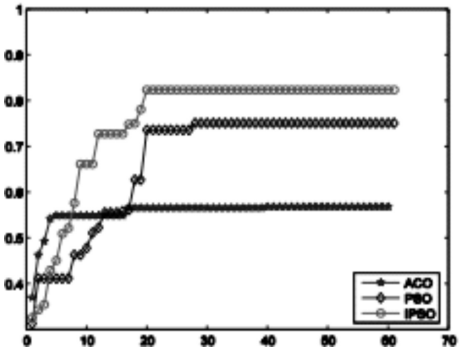

图 3-8 三种算法下最优解的变化曲线(800)

表 3-2 迭代次数为 500 时三种算法下找到最优联盟值时程序运行时间之比

实验环境	任务数	运行时间(秒)		
		IPSO	PSO	ACO
1	4	0	0	0
	12	0	0	0
	30	0	0	0
2	4	5.12	6.34	7.55
	12	10.33	12.02	13.47
	30	32.01	36.52	43.21
3	4	3.06	4.27	8.01
	12	9.53	11.27	17.18
	30	35.26	41.69	45.36

为便于分析表 3-1 和表 3-2，图 3-9 和图 3-10 列出了三种算法下在不同实验环境中联盟值大小及在找到最优联盟值所花费的时间成本对比图。

（下转第 53 页）

以综合指数和法求得各评价对象的综合指标值，该值作为评价对象适宜性等级划分的依据，对每个评价对象得分的情况进行统计分析，根据总频率曲线法在数值的突变处划分各评价对象的适宜等级。一等级评价对象表示开发潜力较强，可以直接开发使用，限制性较少甚至没有。开发、复垦或整理后农作物的产量高，正常利用情况下不会给土地及周边生态环境带来不良后果；二等级评价对象表示开发潜力一般，需要简单改造即可开发使用，具有一定的限制性。开发、复垦或整理后农作物的产量中等，在利用不当情况下将给土地及周边生态环境带来一定的影响；三等级评价对象表示开发潜力较弱，需要采取复杂的工程或生物措施才能开发使用，具有较多的限制。开发、复垦或整理后农作物的产量不高，利用不当时，将给土地及周边生态环境带来严重的影响。

参考文献：

[1] 国家发展改革委. 国家粮食安全中长期规划纲要(2008—2020 年)[Z].北京,2008-11-13.

[2] 刘灵.干旱绿洲区耕地后备资源适宜性评价与开发利用研究—以高台县为例[D].甘肃兰州:西北师范大学,2012.5.

[3] 张雁. 基于 RS、GIS 的喀斯特地区土地适宜性评价研究[D].北京林业大学硕士论文,2007:21.

[4] 黄小清.我国可耕地资源资产评估的理论与方法探讨[J].地理学与国土研究,1996,(1):11-16.

[5] 胡志,朱敖荣,秦侠.特尔菲方法在筛选中国农村初级卫生保健指标体系中的应用.中国卫生统计,1990,7(6):6-9.

[6] 魏华. 土地适宜性评价的物元可拓方法与实证研究[D].广西大学硕士论文,2006:13-14.

(上接第 49 页)

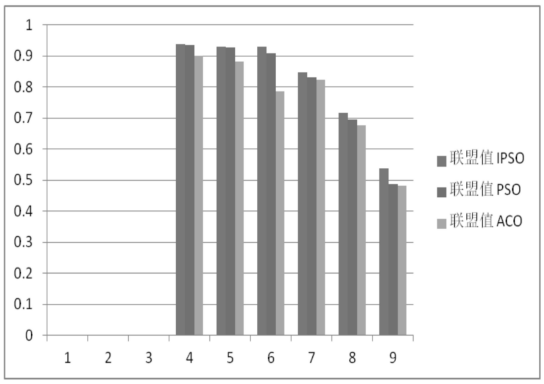

图 3-9 不同算法和实验环境下联盟值比较

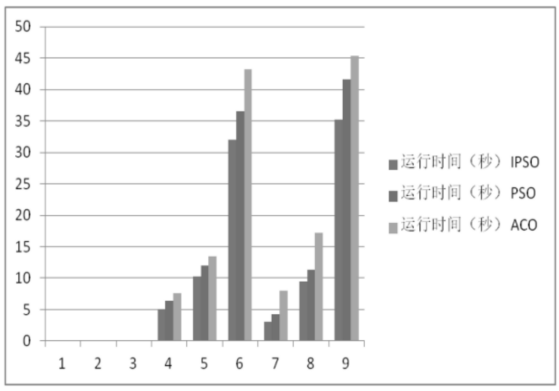

图 3-10 三种算法找到最优联盟值时时间比较

4.2 实验分析

图 3-9 和图 3-10 分别列出了 IPSO、PSO 及 ACO 算法在三种不同实验环境下的联盟最优值及在找到最优联盟值所花费的时间成本对比图，从图中可得出不同算法及环境下只有改进的 IPSO 算法的联盟值最好并且所花费的时间成本最少。从图 3-5- 图 3-10 及表 3-1、表 3-2 中可看出：在实验环境 1 下三种算法都不能完成任务；实验环境 2 下三种算法都可组成联盟，但联盟值是不同的，且相比于 PSO 算法和 ACO 算法本文提出的 IPSO 算法联盟值最优且收敛性最快资源利用率最高；在实验环境 3 下 PSO 和 ACO 算法相比于 IPSO 算法存在资源的极大浪费且联盟值小。除此之外 IPSO 算法在局部极小值方面可看出有所下降且在找到最优解所耗费的时间成本也优于 PSO 和 ACO 算法。综上可知，本文提出的算法无论在联盟值大小、收敛性还是资源利用率上均优于 PSO 算法和 ACO 算法。

5 结束语

多 Agent 联盟是 MAS 中的一个紧迫性问题。基于此文中提出一种改进的 IPSO 算法来解决该问题，同时为克服粒子过早收敛和局部优化，在改进的惯性权重基础上引入一种柯西变异的扰动算子。经验证该 IPSO 算法的全局搜索能力较高，有效地避免了粒子过早收敛，资源浪费等问题。

参考文献：

[1] 李天文. 面向多 Agent 系统的博弈联盟形成与分配问题研究[D].云南大学,2013.

[2] 周昕,凌兴宏. 遗传算法理论及技术研究综述[J]. 计算机与信息技术,2010,04:37-39+43.

[3] 吴庆洪,张颖,马宗民. 蚁群算法综述[J]. 微计算机信息,2011,03:1-2+5.

[4] 蒋建国,夏娜,齐美彬,木春梅. 一种基于蚁群算法的多任务联盟串行生成算法[J]. 电子学报,2005,12:2178-2182.

[5] Kenney J,Eberhart R C. A Discrete Binary Version of the Particle Swarm Algorithm [C] . Proc of the IEEE International Conference on Systems,Man and Cybernetics. Orlando, US, 1997: 4104—4108.