# 一种基于蚁群算法的多任务联盟串行生成算法

蒋建国, 夏 娜, 齐美彬, 木春梅
(合肥工业大学计算机与信息学院, 安徽合肥 230009)

摘 要: 联盟生成是多 Agent 系统的一个关键问题, 主要研究如何在多 Agent 系统中动态生成面向任务的最优 Agent 联盟. 引入蚁群算法解决多任务联盟问题. 提出了一种基于蚁群算法的多任务联盟串行生成算法, 对于任务序列可依次生成全局最优联盟, 避免了联盟死锁和资源浪费, 同时算法基于蚁群系统的学习能力可以有效减少联盟生成的搜索时间和计算量, 可实现性好.

关键词: 多 Agent 系统; 联盟; 蚁群算法; 信息素
中图分类号: TP18 文献标识码: A 文章编号: 0372-2112 (2005) 12-2178-05

# An Ant Colony Algorithm Based Multi-task Coalition Serial Generation Algorithm

JIANG Jian-guo, XIA Na, QI Mei-bin, MU Chun-mei
(*School of Computer & Information, Hefei University of Technology, Hefei, Anhui 230009, China*)

**Abstract:** Coalition Generation is a key topic in Multi-Agent System. It mainly researches how to generate the optimal task-oriented coalition in dynamic manner. This paper adopts Ant Colony Algorithm to solve the problem, and presents an Ant Colony Algorithm based multi-task coalition serial generation algorithm. This method can generate the optimal coalitions one after another for the task alignment, avoid the coalition locking and resource wasting, and the learning ability of ant colony system can reduce the searching time and computing works effectively.

**Key words:** multi-agent system (MAS); coalition; ant colony algorithm; pheromone

## 1 引言

Agent 间通过组成联盟, 可以提高求解问题的能力, 获得更多的报酬, 因此联盟是多 Agent 系统(MAS)的重要合作方法. 从 1993 年提出联盟方法以来, Shehory 和 Kraus[1], 以及 Sandholm 和 Lesser 作了大量工作[2,3]. 近年来, 联盟生成已成为多 Agent 系统研究的一个重要方面, 联盟生成问题可形式化描述如下:

设 Agent 集 $N=\{A_1, A_2, \cdots, A_n\}$, 任意 $A_i$ 都有一个能力向量: $B_i=\langle b_i^1, b_i^2, \cdots, b_i^r\rangle$, $b_i^j \geqslant 0$, $(1\leqslant i\leqslant n, 1\leqslant j\leqslant r)$, 用于定量描述 $A_i$ 执行某种特定动作的能力大小. 设任务集 $T=\{t_1, t_2, \cdots, t_m\}$, 每个任务 $t_j$ 均有一定的能力需求 $B_j=\langle b_j^1, b_j^2, \cdots, b_j^r\rangle$, $b_j^k \geqslant 0$, $(1\leqslant j\leqslant m, 1\leqslant k\leqslant r)$, Agent 完成任务 $t_j$ 可获得相应的利益 $P(t_j)$.

我们定义一个联盟 $C$ 是 $N$ 的一个非空子集. 联盟 $C$ 有一个能力向量 $\boldsymbol{B}_C=\langle b_C^1, b_C^2, \cdots, b_C^r\rangle$, $\boldsymbol{B}_C$ 是联盟中所有 Agent 能力向量的总和, 即 $B_C=\sum_{A_i\in C} B_i$. 联盟 $C$ 可以完成任务 $t_j$ 的必要条件是: $\forall 1\leqslant i\leqslant r$, $b_j^i\leqslant b_C^i$.

我们作以下假设: (1) 和惯例一样[2~5], 在特征函数对策 (character function games, CFGs) 中研究联盟形成. 每个联盟 $C$ 的值用一个特征函数 $V(C)$ 给出. 我们假定 $V(C)\geqslant 0$, $V(C)=P(t_j)-F(C)-C(C)$. 式中 $P(t_j)$ 指完成任务 $t_j$ 所获得的利益; $F(C)$ 指联盟成员总能力折合的成本; $C(C)$ 指联盟协作求解 $t_j$ 过程中的额外开销, 主要是通信开销. 如果联盟 $C$ 不满足上述必要条件, 则 $V(C)$ 为 0, 否则 $V(C)$ 为正数. (2) 在非超加性环境中研究联盟形成[3] (超加性是指 $\forall C_1, C_2\subseteq N$, $C_1\cap C_2=\Phi$, 有 $V(C_1\cup C_2)\geqslant V(C_1)+V(C_2)$, 在这样的环境中, 最大的联盟将是最有益的).

多任务联盟生成问题就是对于任务序列 $t_1, t_2, \cdots, t_m$, 寻找 $m$ 个联盟, 使得联盟值总和尽可能大. 这是一个复杂的组合优化问题.

## 2 相关的工作分析

Sandholm[2,3]、Sen[4]、胡山立[6]、骆正虎[7] 均基于"联盟结构"解决多任务联盟问题:

(1) 定义一个联盟结构 CS 是 Agent 集的一个划分, 从而同一个联盟结构中的联盟是不相交的. 联盟结构 $CS=\{C_1, \cdots, C_m\}$ 中的每个 $C_j$ 对应一个 $t_j$, 若对应于任务 $t_l$ 的联盟 $C_l$ 不存在, 则联盟结构不包含任何成员. 联盟结构的值是该结构中所

收稿日期: 2004-11-05; 修回日期: 2004-11-22
基金项目: 国家自然科学基金(No. 60474035)

有联盟值的总和,即 $V(CS)=\sum_{C\in CS}V(C)$. 所有联盟结构的集合为 $M$.

(2) 把多任务联盟问题看作是在联盟结构图中搜索,寻找能极大化联盟值总和的联盟结构 $CS^*$,即寻找最优的 Agent 集合划分形式. 这些算法大多通过部分搜索找到一个能保证其联盟结构值与最优的联盟结构值相距在一个有限的界限内的联盟结构.

性能分析:

(1) 方法的及时性、有效性. 当 $B_{all\_Agents}=\sum_{A_i\in N}B_i< B_{m\_tasks}=\sum_{t_j\in T}B_j$, 对于 $\forall CS\in M$, $\exists t_l$ 无法完成,联盟结构不包含任何成员,出现死锁;而当 $B_{all\_Agents}\gg B_{m\_tasks}$, 对于 $\forall CS\in M$, 因为 $F(CS)$ 和 $C(CS)$ 一定且极大,所以 $V(CS)$ 甚小.

(2) 计算的可实现性. 因为此类算法没有考虑联盟形成的经验学习,在任务完成后联盟解体,每当新的任务加入任务集时,需要依生成算法重新结盟,这样势必有大量的计算和通信开销. 这对计算资源受限的多 Agent 系统是难以实现的.

针对这些问题,已有学者提出联盟演化机制[8],在求解等价联盟问题时计算量明显减少,但在匹配不成功的情况下调整方法复杂,易失效.

本文在参考这些成果的基础上,引入蚁群算法来解决多任务联盟生成问题,提出了一种基于蚁群算法的多任务联盟串行生成算法,对于任务序列可以依次生成全局最优联盟,同时算法基于蚁群系统的学习能力比基于等价的联盟演化机制灵活、有效、有更高的鲁棒性,可以有效减少联盟生成的搜索时间和计算量. 文章组织如下:首先是算法设计,包括基于基本蚁群算法求解单任务联盟和多任务联盟、算法的个性化和改进以及算法的整体描述,然后给出了相关算法的比较和分析结论.

## 3 算法

### 3.1 蚁群算法

蚁群算法(ant colony algorithm, ACA)是一种新型的模拟进化算法. 它是在对自然界中真实蚁群的集体行为的研究基础上,由意大利学者 M. Dorigo[9]等首先提出的.

蚁群算法模拟真实蚁群的协作过程,算法由许多蚂蚁共同完成. 每只蚂蚁在候选解的空间中独立地搜索解,并在所寻得的解上留下一定的信息量. 解的性能越好蚂蚁留在其上的信息量越大,信息量越大的解被选择的可能性也越大. 在算法的最初阶段所有解上的信息量是相同的,随着算法的推进较优解上的信息量增加,算法渐渐收敛. 蚁群算法已成功解决了一系列问题,如TSP 问题、分配问题、Jobr shop 问题,初步研究已显示出蚁群算法在求解复杂组合优化问题方面具有并行化、正反馈、鲁棒性强等先天优越性.

### 3.2 基于基本蚁群算法求解单任务联盟

在含有 $n$ 个 Agent 的MAS 中,寻找能够完成任务 $t_j$ 且联盟值最大的Agent 联盟 $C$. 我们借助蚁群算法,让蚂蚁选择以前合作过且合作效果较好(联盟值较大)的 Agent 再次结成联盟,并以正反馈的方式,逐渐形成联盟值最大或近似最大的联盟 $C$.

首先引入如下记号:设 $m$ 是蚁群中蚂蚁的数量,$d_{ij}(i,j=1,2,\cdots,n)$ 表示 Agent $i$ 和 Agent $j$ 之间的距离,即通信开销;$\tau_{ij}(t)$ 表示 $t$ 时刻在 $ij$ 连线上残留的信息素量,物理意义为"熟悉度"或"友好度". 初始时刻,各条边上的信息素量相等,设 $\tau_{ij}(0)=\tau_0$. 蚂蚁 $k(k=1,2,\cdots,m)$ 在运动过程中,根据各条边上的信息素量决定转移方向,每到一个节点就将该 Agent 加为盟友,$p_{ij}^k$ 表示在 $t$ 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 Agent $j$ 加入联盟的概率,

$$p_{ij}^k=\begin{cases}\dfrac{[\tau_{ij}(t)]^{\alpha}[1/d_{ij}]^{\beta}}{\sum_{u\in J_k}[\tau_{iu}(t)]^{\alpha}[1/d_{iu}]^{\beta}}, & j\in J_k\\ 0, & \text{其他}\end{cases}\tag{1}$$

其中,$J_k$ 为蚂蚁 $k$ 目前尚未访问过的Agent 集合. 参数 α、β 分别用来控制熟悉度和通信开销的相对重要程度.

当蚂蚁到达某个节点发现当前的 Agent 联盟已可以完成任务 $t_j$ 时,即停止寻径. 这就要求蚂蚁每到一个节点就累加该 Agent 能力向量,计算出当前联盟能力向量,并判断是否已达到任务的能力需求,若是,则停止寻径,否则继续寻径. 当最后一只蚂蚁形成任务求解联盟并停止寻径时,一次循环结束.

随着时间的推移,以前 Agent 之间的熟悉度逐渐降低(熟人渐忘),用参数 1-ρ 表示熟悉度的降低程度,蚂蚁们完成一次循环,各条边上的熟悉度根据下式进行调整:

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

$\Delta\tau_{ij}^k$ 表示第 $k$ 只蚂蚁在本次循环中对 Agent $i$、$j$ 之间熟悉度的增量.

$$\Delta\tau_{ij}^k=\begin{cases}\dfrac{V(C_k)}{\sum_{k=1}^{m}V(C_k)}, & \text{如果蚂蚁 } k \text{ 形成的联盟包含 Agent.i.j}\\ 0, & \text{其他}\end{cases}\tag{3}$$

其中,$V(C_k)$ 是蚂蚁 $k$ 形成的联盟的值. 这里对熟悉度的调整利用的是整体信息,在求解 Agent 联盟问题时性能较好. 基本蚁群算法中参数 α、β、ρ 用实验方法确定其最优组合. 停止条件用固定进化代数或者当进化趋势不明显时便停止计算. 算法复杂度为 $O(NC\cdot n^3)$,其中,$NC$ 表示循环次数.

### 3.3 求解多任务联盟

首先根据紧迫度 $U(t_j)$ 对 $T$ 中的任务进行排序,然后依次求解. 当算法求得任务 $t_j$ 的最优 Agent 联盟开始求解下一个任务 $t_{j+1}$ 的最优联盟时,Agent 之间的信息素不再是初始值 $\tau_0$,而是上次求解结束时残留的信息素 $\tau_{ij}(t)$. 它作为一种非常宝贵的经验知识,将有效指导蚁群算法后面的求解过程,减少搜索时间和计算量.

可能出现的情况是:在算法生成前两个任务的联盟时,计算量较大,但随着联盟历史的积累,后面联盟的生成速度越来越快,计算量越来越小. 这么说是有根据的,因为系统待求解任务的性质、类型不会发生根本性的变化(显然这是符合实际

情况的),这就意味着:系统经过一段时间的运行后,很多 Agent 可能会形成相对稳定的并且是成功的联盟.在任务发生重大变化的情况下,算法仍然可以找到最优解,不过搜索时间要长一些,因为此时的先验知识失去了指导意义.

### 3.4 算法的个性化与改进

用蚁群算法求解最优 Agent 联盟时,同样存在着收敛慢、易于陷入局部最小等缺陷.为了提高算法的全局搜索能力和搜索速度,我们对蚁群算法作以下改进:

(1)每次循环开始,重新随机放置 $m$ 只蚂蚁到 $n$ 个 Agent 上.这样可以给更多 Agent 参加联盟的机会,有助于解的多样化,提高蚁群算法的全局搜索能力.

(2) $t$ 时刻蚂蚁 $k$ 选择 Agent $j$ 加入联盟的概率与当前联盟所有成员有关,

$$p_{ij}^{k}=\begin{cases}\dfrac{[\tau_{1j}(t)+\tau_{2j}(t)+\cdots+\tau_{ij}(t)]^{\alpha}[1/(d_{1j}+d_{2j}+\cdots+d_{ij})]^{\beta}}{\sum_{u\in J_k}[\tau_{1u}(t)+\tau_{2u}(t)+\cdots+\tau_{iu}(t)]^{\alpha}} \\ [1/(d_{1u}+d_{2u}+\cdots+d_{iu})]^{\beta}, j\in J_k \\ 0, \text{其他}\end{cases} \tag{4}$$

(3)引入第二种信息素"内激素",在出现局部极小情况时通过内激素的作用使解跳出局部极小,从而向最优解方向继续进化.

定义 内激素 $R=0.9^{g}$ 是对蚂蚁自身的思维状态、价值取向进行调整,其作用形式为:

$$p_{ij}^{k}=\begin{cases}\dfrac{[\tau_{(1,2,\cdots,i)j}(t)]^{\alpha R}[1/(d_{(1,2,\cdots,i)j})]^{\beta}}{\sum_{u\in J_k}[\tau_{(1,2,\cdots,i)u}(t)]^{\alpha R}[1/d_{(1,2,\cdots,i)u}]^{\beta}}, j\in J_k \\ 0, \text{其他}\end{cases} \tag{5}$$

$$g=\begin{cases}0 & \text{算法求得的最优解仍在进化} \\ g+1, & \text{最优解在 } N \text{ 次循环内没有明显改进,} \\ & \text{且 } g+1\leqslant g_{\max} \\ g_{\max}, & \text{其他}\end{cases} \tag{6}$$

$g$ 是内激素的代数,每次循环结束,根据求得的最优解的情况对 $g$ 进行调整.

可见,在蚁群系统运行初期,算法求得的最优解仍在进化时,内激素 $R=1$,不影响蚂蚁对 Agent 的选择概率;而当算法求得的最优解在 $N$ 次循环内没有明显改进,即出现可能的局部极小情况时,内激素开始作用,显著降低信息素在选择概率 $p_{ij}^{k}$ 中的重要程度,相应提升启发式信息的受重视程度,从而蚂蚁选择以前选过的 Agent 的可能性降低,倾向于探索新解;在最优解仍没有改进的情况下内激素会加倍发挥作用,增大扰动,使解更易跳出局部极小;另外,在提高了算法的全局搜索能力的同时为了保证算法的收敛速度,我们引入 $g_{\max}$ 控制内激素的强度.一旦解跳出局部极小,算法求得的最优解又开始进化时,则 $g=0$, $R=1$,内激素又不起作用,开始进入"潜伏期".

### 3.5 算法描述

应用个性化和改进的蚁群算法以串行方式求解多任务联盟问题,具体的算法伪代码如下:

Step 0. Set $t=0$,每条边上的 $\tau_{ij}(0)=\tau_0$

Step 1. If $(T\neq\Phi)$
then 根据任务的紧迫度 $U(t_j)$,对 $T$ 中的任务进行排序 $t'_1, t'_2, \cdots t=t'_1$(当前最紧急的任务)
else 终止整个程序

Step 2. If $(B_{\text{all\_Agents}}<B_t)$ then Goto step 10
else {初始化:Set $NC=0$, $\Delta\tau_{ij}=0$, $g=0$, $J_k=(1,2,\cdots,n)$}

Step 3. 随机放置 $m$ 只蚂蚁到 $n$ 个 Agent 上
For $k=1$ to $m$ do
把第 $k$ 个蚂蚁所在的初始 Agent 号码从 $J_k$ 中删除,计算初始联盟能力向量 $B_{C_k}$

Step 4. For $k=1$ to $m$ do
While$(B_{C_k}<B_t)$
{根据式(5)的概率 $p_{ij}^{k}$ 选择下一个 Agent,将第 $k$ 个蚂蚁移到 Agent $j$,并将 $j$ 从 $J_k$ 中删除,计算当前联盟能力向量 $B_{C_k}$}

Step 5. For $k=1$ to $m$ do
根据式 $V(C_k)=P(t_j)-F(C_k)-C(C_k)$ 计算第 $k$ 个蚂蚁形成的联盟的值 $V(C_k)$,更新最大联盟值及其联盟.

Step 6. If(最大联盟值== $N$ 个循环以前的最大联盟值)
then If $(g++<=g_{\max})$ then $g++$
else $g=g_{\max}$

Step 7. For $k=1$ to $m$ do
根据式(2)、(3)更新边上的熟悉度 $\tau_{ij}(t+1)$

Step 8. Set $t=t+1$
Set NC= NC+1
Set $\Delta\tau_{ij}=0$

Step 9. If (NC< NC) and (不是长时间无进化) then $J_k(1,2,\cdots,n)$, Goto step 3 else 输出最大联盟值及其联盟

Step 10. 将 $t$ 从 $T$ 中删除, Goto step 1

## 4 与相关算法比较

为了检验算法性能,将本文的算法与文献[2]、文献[7]的算法进行比较.首先产生实验所需的参数:Agent 数 $n=20$、任务数 $m=4$、每个 Agent 的能力向量 $B_i$、Agent 之间的通信开销 $d_{ij}$、任务的 $t_j(1\leqslant j\leqslant 4)$ 能力需求 $B_{t_j}$;然后在四种典型环境下分别应用三种算法求解相同的多任务联盟问题,并对所得结果进行了比较.

文献[7]的遗传算法采用一维自然数编码,种群规模设为 20、交叉概率 $p_c$ 设为 0.95、变异概率 $p_m$ 为 0.02.在本文算法中,设置蚂蚁数 $m=10$、Agent 之间的初始熟悉度 $\tau_{ij}(0)=\tau_0=1$、$\alpha=4$、$\beta=1$、$\rho=0.9$, $NC_{\max}=5000$;内激素 $R=0.9^{g}$ 中的常数 0.9 是经验数值,大量实验表明该数值在 0.85~0.95 之间时,内激素对算法的改进效果较好;另设参数 $g_{\max}=5$ 和 $g_{\max}=10$ 两种情况.

环境 **1** $B_{\text{all\_Agents}}=\sum_{A_i\in N}B_i<B_{t_j}$, $(1\leqslant j\leqslant 4)$,三种算法都无法生成任务求解联盟.

环境 **2**　$(B_{all\_Agents} \geqslant B_{t_j}, (\exists\, 1 \leqslant j \leqslant 4)) \wedge (B_{all\_Agents} < B_{m\_tasks} = \sum_{t_j \in T} B_{t_j})$, 文献[2]和文献[7]的算法因为联盟死锁, 仍无法生成任务求解联盟; 应用本文算法, 对于所有 $t_j$ ($B_{t_j} \leqslant B_{all\_Agents}$), 均可生成相应的全局最优联盟.

环境 **3**　$B_{all\_Agents} \geqslant B_{m\_tasks}$, 三种方法都可以生成任务求解联盟, 但结果的质量有差异(见表 1), 本文算法生成的联盟其联盟值总和最大, 结果最优.

环境 **4**　$B_{all\_Agents} \gg B_{m\_tasks}$, 文献[2]和文献[7]的算法因为极大的资源浪费, 所以联盟值甚小; 而本文算法所得结果的优势更明显.

表 1 给出了在四种环境下三种算法求解多任务联盟的实验结果比较; 图 1 示出了本文算法在环境 3 中所求解的进化过程曲线.

表 **1**　三种算法的性能比较

| | 文献[2]的算法 | | | 文献[7]的算法 | | | 本文算法($g_{max}=5$) | | |
|---|---|---|---|---|---|---|---|---|---|
| 环境 1 | 无解 | | | 无解 | | | 无解 | | |
| 环境 2 | 无解 | | | 无解 | | | $V(C_1)$ | 0 | $\sum V(C_j)$ = 1131.5 |
| | | | | | | | $V(C_2)$ | 0 | |
| | | | | | | | $V(C_3)$ | 673.3 | |
| | | | | | | | $V(C_4)$ | 458.2 | |
| 环境 3 | $V(C_1)$ | 325.3 | $\sum V(C_j)$ = 715.4 | $V(C_1)$ | 315.1 | $\sum V(C_j)$ = 733.3 | $V(C_1)$ | 497.5 | $\sum V(C_j)$ = 1047.5 |
| | $V(C_2)$ | 131.7 | | $V(C_2)$ | 158.7 | | $V(C_2)$ | 197.9 | |
| | $V(C_3)$ | 146.0 | | $V(C_3)$ | 150.3 | | $V(C_3)$ | 187.5 | |
| | $V(C_4)$ | 112.4 | | $V(C_4)$ | 109.2 | | $V(C_4)$ | 164.6 | |
| 环境 4 | $V(C_1)$ | 30.9 | $\sum V(C_j)$ = 73.5 | $V(C_1)$ | 29.3 | $V(C_j)$ = 72.2 | $V(C_1)$ | 95.2 | $\sum V(C_j)$ = 280.8 |
| | $V(C_2)$ | 18.0 | | $V(C_2)$ | 18.6 | | $V(C_2)$ | 68.4 | |
| | $V(C_3)$ | 13.4 | | $V(C_3)$ | 14.5 | | $V(C_3)$ | 65.3 | |
| | $V(C_4)$ | 10.7 | | $V(C_4)$ | 9.8 | | $V(C_4)$ | 51.9 | |

从表 1 可以看出, 在后三种典型环境下本文算法可以及时、高效地生成多任务多个联盟, 解的质量明显优于前两种算法, 避免了联盟死锁和资源浪费; 由图 1 可见, 在计算可实现性方面, 本文算法在生成第一个任务的联盟时, 搜索时间较长, 计算量较大, 但随着联盟历史的积累, 后面联盟的生成越来越快, 计算量越来越小, 可实现性较好.

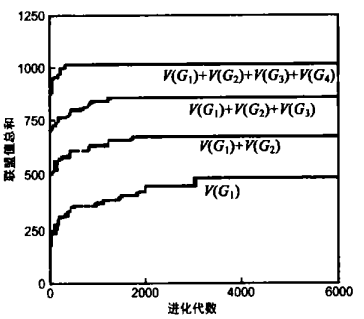



**图 1　本文算法在环境 3 中所求解的进化过程曲线**

## 5　结论

本文引入蚁群算法来解决多任务联盟生成问题, 提出了一种基于蚁群算法的多任务联盟串行生成算法, 以串行方式依次生成多任务的全局最优联盟, 不仅避免了联盟死锁和资源浪费, 而且解的质量明显优于相关算法; 同时算法基于蚁群系统的学习能力比基于等价的联盟演化机制灵活、有效、有更高的鲁棒性, 可以有效减少联盟生成的搜索时间和计算量, 可实现性较好.

参考文献:

[1] O Shehory, S Kraus. Task allocation via coalition formation among autonomous agents[A]. Proc of IJCAI-95[C]. Los Arrgeles, CA, USA: Morgan Kaufmann Publishers, 1995. 655-661.

[2] T Sandholm, K Larson, M Andersson, et al. Anytime coalition structure generation with worst case guarantees[A]. Proc of the National Conference on Artificial Intelligence[C]. Madison, WI, 1998. 46-53.

[3] T Sandholm, V Lesser. Coalition among computationally bounded agents[J]. Artificial Intelligence, 1997, 94(1): 99-137.

[4] S Sen, P S Dutta. Searching for optimal coalition structures[A]. Proc of the 4th ICMAS[C]. Boston, MA, USA: IEEE Press, 2000. 287-292.

[5] 蒋建国, 夏娜, 于春华. 基于能力向量发挥率和拍卖的联盟形成策略[J]. 电子学报, 2004, 32(12A): 215-217.
JIANG Jiarr guo, XIA Na, YU Churr hua. The Coalition Formation Strategy Based on Capability Vector Contributiorr Rate and Auction[J]. Acta Electronica Sinica, 2004, 32(12A): 215-217. (in Chinese)

[6] 胡山立, 石纯一. 一种任意时间联盟结构生成算法[J]. 软件学报, 2001, 12(5): 729-734.
HU Sharr li, SHI Churr yi. An Anytime coalition structure generation algorithm[J]. Journal of Software, 2001, 12(5): 729-734. (in Chinese)

[7] 骆正虎. 移动 Agent 系统若干关键技术问题研究[D]. 合肥: 合肥工业大学研究生院, 2002. 81-98.
LUO Zheng-hu. Research on several key problems in mobile agent systems[D]. Hefei: Institute of postgraduate, Hefei University of Technology, 2002. 81-98. (in Chinese)

[8] 徐晋辉, 石纯一. 一种基于等价的联盟演化机制[J]. 计

算机研究与发展, 1999, 36(5): 513- 517.
XU Jin-hui, SHI Chun-yi. A mechanism of coalition evolvement based on equivalence[J]. Journal of Computer Research & Development, 1999, 36(5): 513- 517. (in Chinese)

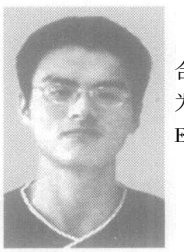
[9] M Dorigo, V Maniezzo, A Colorni. The ant system: optimization by a colony of cooperating agents[J]. IEEE Transactions on Systems, Man, and Cybernetics-Part B, 1996, 26(1): 29-41.

[10] 王颖, 谢剑英. 一种自适应蚁群算法及其仿真研究[J]. 系统仿真学报, 2002, 14(1): 31- 33.
WANG Ying, XIE Jian-ying. An adaptive ant colony optimization algorithm and simulation[J]. Journal of System Simulation, 2002, 14(1): 31- 33. (in Chinese)

[11] 蒋建国, 骆正虎, 张浩, 等. 基于改进型蚁群算法求解旅行 Agent 问题[J]. 模式识别与人工智能, 2003, 16(1): 6- 12.
JIANG Jian-guo, LUO Zheng-hu, ZHANG Hao, et al. Solution to traveling agent problem based on improved ant colony algorithm[J]. Pattern Recognition and Artificial Intelligence, 2003, 16(1): 6- 12. (in Chinese)

作者简介:

蒋建国 男, 1955 年 10 月出生于安徽黄山, 合肥工业大学教授, 博士生导师, 主要研究方向为分布式智能控制系统、数字图像分析与处理、DSP 技术与应用等. E-mail: jjg@ahl65.net.



夏 娜 男, 1979 年 10 月出生于安徽芜湖, 合肥工业大学讲师, 在职博士生, 主要研究方向为分布式人工智能、智能控制、计算智能等.
E-mail: xiananawo@tom.com.

齐美彬 男, 1969 年 7 月出生于安徽东至, 合肥工业大学副教授, 主要研究方向为信号与信息处理、计算机控制等.

木春梅 女, 1979 年 4 月出生于安徽涡阳, 合肥工业大学讲师, 主要研究方向为数字图像处理、人工智能等.