文章编号: 1000−8152(2011)08−1071−10

可控搜索偏向的二元蚁群算法

胡　钢, 熊伟清, 张　翔, 袁军良

(宁波大学 电子商务研究所, 浙江 宁波 315211)

摘要: 蚁群算法按照信息素轨迹产生的偏向对解空间进行搜索. 当前改进蚁群算法性能的主要方法是提高种群的多样性, 少有对搜索偏向进行控制. 本文以可控搜索偏向作为研究的出发点, 通过对至今最优信息素更新方式的分析, 得出了从任意代到算法收敛没有发现较优解的概率下限. 并以此为基础, 把访问量与蚂蚁数量的关系作为控制偏向的依据, 在兼顾提高种群多样性的前提下, 设计了可控搜索偏向的二元蚁群算法. 通过多个函数的测试以及0−1多背包问题的应用, 其实验结果表明该算法有较好的搜索能力以及较快的收敛速度.

关键词: 蚁群算法; 二元蚁群算法; 信息素更新方式; 可控搜索; 函数优化; 0−1多背包问题

中图分类号: TP18　　**文献标识码:** A

Binary ant colony algorithm with controllable search bias

HU Gang, XIONG Wei-qing, ZHANG Xiang, YUAN Jun-liang

(Institute of Electronic Commerce, Ningbo University, Ningbo Zhejiang 315211, China)

Abstract: Ant colony algorithm explores the solution space according to the bias produced by pheromone trail. However, most of the existing improvements concentrate in raising the population diversity, instead of controlling the search bias. On the basis of the controllable search bias and by the update pattern of the current pheromone, we determine for any given iteration the lower bound of the probability of no further improvement in solution up to the convergence. Using the relation between the number of visitors and the ant population, and considering the population diversity, we develop a binary ant colony algorithm with controllable search bias. In the test of function optimization and the application to the 0−1 multiple knapsack problem, the algorithm exhibits a good search ability and a high convergence speed.

Key words: ant colony algorithm; binary ant colony algorithm; pheromone update pattern; controllable search; function optimization; 0−1 multiple knapsack problem

1　引言(Introduction)

群智能(swarm intelligence)是一种相对较新的求解问题的方法, 它起源于对昆虫或其他动物社会行为的模仿. 受蚁群在觅食过程中总能找到从巢穴到食物源的最短路径这一现象的启发, M. Dorigo提出了用于求解旅行商问题(traveling salesman problem, TSP)的蚂蚁系统(ant system, AS)[1], 后来又归纳提炼出蚁群优化(ant colony optimization, ACO)[2]的元启发方法. 蚁群优化算法最初用于求解调度问题、路由问题、时间表问题等组合优化问题. 随着蚁群优化算法的发展, 在连续域上也得到了广泛的应用.

在文献[3]中, 将蚁群算法与二进制编码方式相结合, 提出了二元蚁群进化算法, 并成功应用于连续优化问题和离散优化问题. 与标准蚁群算法相比, 虽然二元蚁群算法效率高、求解速度快, 但仍存在易陷入局部解的缺点. 作为改进, 文献[4]提出了拥塞控制策略, 使得二元蚁群算法性能有了较大的提高.

蚁群算法以及其他一些进化算法在搜索过程中都是按照某个搜索偏向(search bias)进行搜索, 一旦算法产生错误的搜索偏向, 则使搜索过程陷入局部最优. 而当前对算法的改进仍然以提高群体的多样性为主要方法, 少有对搜索偏向进行控制. 本文通过对至今最优信息素更新方式的分析, 得出从任意代到算法收敛没有发现较优解(定义4)的概率下限. 在此基础上, 把访问量与蚂蚁数量的关系作为控制偏向的依据, 在包含提高种群多样性的思想下, 设计了可控搜索偏向的二元蚁群算法.

2　蚁群优化算法的基本框架(The basic framework of ACO algorithm)

二元蚁群算法将解空间与搜索空间分离, 其搜索空间为n维离散空间, 在本质上可以理解为求解组合优化问题(combinatorial optimization, CO), 对于这类问题可以形式的定义为:

定义 1[5,6]　组合优化问题可由一个模型P =

收稿日期: 2009−12−01; 收修改稿日期: 2010−11−15.

基金项目: 浙江省自然科学基金资助项目(Y1080364); 宁波市自然科学基金资助项目(Y1100052).

(S,Ω,f)来定义. 其中: S是定义在一组有限的离散决策变量上的可行解空间, Ω为决策变量之间的约束条件的集合, $f:S\to\mathbb{R}^+$为一个目标函数. 优化问题就是要在S中找出满足约束条件Ω的使f最小的解s^*.

可行解空间S定义为: 给定n个离散的决策变量$X_i(i=1,2,\cdots,n)$, 其中X_i的定义域为$D_i=\{c_i^1, c_i^2, c_i^{|D_i|}\}$, 对这些决策变量的一组符合约束条件的赋值$X_i=c_i^j(i=1,2,\cdots,n)$, 决定了一个可行解$s$, S即为所有这些可行解s的集合.

如果约束条件集合Ω为空, 每一个决策变量可以独立地取定义域中的任意值, 称P为一个无约束的优化问题模型, 否则为约束优化问题模型. 如果存在可行解$s^*\in S$, 使得$f(s^*)\leqslant f(s),\forall s\in S$, 则称$s^*$为全局最优解, 所有全局最优解的集合记为$S^*\subseteq S$. 求解优化问题就是找出$S^*$中的一个或全部解。

在蚁群算法中, 对每一个解分量值c_i^j定义一个信息素τ_i^j, 所有的信息素取值的集合构成信息素模型τ. 蚁群算法中的每只蚂蚁搜索合适的解分量值, 以构造一个可行解. 设S^P为蚂蚁已经搜索到的分量所构成的部分解, 在选取每个决策变量X_i的值时, 蚂蚁选取c_i^j的概率$P(c_i^j|S^P)$定义为

$$P(c_i^j|S^P)=\frac{(\tau_i^j)^\alpha[\eta(c_i^j)]^\beta}{\sum\limits_{c_i^k\in\sigma(S^P)}(\tau_i^k)^\alpha[\eta(c_i^k)]^\beta},\ \forall c_i^j\in\sigma(S^P),\tag{1}$$

式中: $\sigma(S^P)$为对部分解S^P而言, 在约束条件Ω限制下决策变量X_i可取值的集合; $\eta(c_i^j)$是对应解分量值c_i^j的启发式信息; α,β分别为信息素及启发式信息影响程度的参数. 在每次迭代后, 要按下式对信息素进行更新:

$$\tau_i^j(t+1)=(1-\rho)\tau_i^j(t)+g(s),\tag{2}$$

其中: ρ为挥发系数, $i=1,2,\cdots,n$, $j=1,2,\cdots,|D_i|$, $0<g(s)<+\infty$, $g:S\to\mathbb{R}^+$. 为满足下列条件的质量函数: 对$\forall s,s'\in S, s\neq s'$, 若$f(s)<f(s')$则必有$g(s)\geqslant g(s')$.

3 没有发现较优解的概率下限(The probability floor of finding no better solution)

为了清晰起见, 首先做几个定义:

定义 2 只用至今最优解更新信息素的更新规则, 叫做至今最优信息素更新方式.

定义 3 如果$\tau_i^j=\tau_{\min}, \tau_i^k=\tau_{\max}(i=1,2,\cdots,n,\ 1\leqslant k\leqslant|D_i|,\ j=1,2,\cdots,|D_i|$且$k\neq j)$, 称算法已经收敛.

定义 4 如果t代的至今最优解为$\hat{s}(t)$, 有$s_1\in S$, 且$f(s_1)<f(\hat{s}(t))$, 称s_1为t代较优解.

定义 5 从第t代到算法收敛, 没有发现t代较优解的最小概率, 称为t代概率下限, 简写为LPB(t).

在下面的分析中, 将忽略启发式信息, 具体原因请参见文献[7]. 对于至今最优信息素更新方式, 首先考虑一个已有的定理:

定理 1[8] 从第$t'\geqslant t^*+t_0$代开始, 在至今最优信息素更新方式下, 有$\forall(i,\ j)\notin s^*, \tau_i^j=\tau_{\min}$. 其中$t^*$为第一次找到全局最优解$s^*$的代数, 且有

$$t_0=\lceil\frac{\ln\tau_{\min}-\ln\tau_{\max}}{\ln(1-\rho)}\rceil.\tag{3}$$

引理 1 如果t代的至今最优解为$\hat{s}(t)$, $\hat{t}$为第一次找到$\hat{s}(t)$的代数, 则从第$t''\geqslant\hat{t}+t_1$代开始, 在至今最优信息素更新方式下, $\exists(i,j),\ \tau_i^k>\tau_{\min}$, $\tau_i^j=\tau_{\min}(i=1,2,\cdots,n,\ 1\leqslant j\leqslant|D_i|, k=1,2,\cdots,|D_i|, k\neq j)$, 且$t_1\geqslant t_0$.

证 在$\hat{t}+1$代, 若$\hat{s}(t)=\hat{s}(t+1)$, 则$\forall(i,j)\notin\hat{s}(t)$,

$$\tau_i^j(\hat{t}+1)\leqslant\max\{\tau_{\min},(1-\rho)\tau_{\max}\}.$$

同理, 若在$\hat{t}+t_1$代, 仍然有$\hat{s}(t)=\hat{s}(t+t_1)$, 则

$$\tau_i^j(\hat{t}+t_1)\leqslant\max\{\tau_{\min},(1-\rho)^{t_1}\tau_{\max}\}.$$

t_1是第一次使得$(1-\rho)^{t_1}\tau_{\max}\leqslant\tau_{\min}$的代数. 所以很容易得到$t_1=\lceil(\ln\tau_{\min}-\ln\tau_{\max})/(\ln(1-\rho))\rceil$. 如果在$\hat{t}$代以后, 有$f(\hat{s}(t+c))\leqslant f(\hat{s}(t))$, 其中$\hat{s}(t+c)$为$t+c$代所找到的至今最优解, 则从$t+c$代开始, 重复上面的计算过程, 所以有$t_1\geqslant\lceil(\ln\tau_{\min}-\ln\tau_{\max})/\ln(1-\rho)\rceil$. 证毕.

引理 2 如果t代至今最优解为$\hat{s}(t)$, $\hat{t}$为第一次找到$\hat{s}(t)$的代数. 在至今最优信息素更新方式下, $\hat{t}+t_{\rm c}$代时算法必定收敛, $t_{\rm c}\geqslant t_1$.

证 算法收敛不仅需要$\exists(i,k),\tau_i^k=\tau_{\max}$, 且要$\forall(i,j),\tau_i^j=\tau_{\min}(i=1,2,\cdots,n,\ 1\leqslant k\leqslant|D_i|, j=1,2,\cdots,|D_i|, k\neq j)$, 因此$t_{\rm c}=\max\{t_1,t_{\rm m}\}$, 其中$t_{\rm m}$为使得$\tau_i^k=\tau_{\max}$的代数. 根据引理1, 所以有$t_{\rm c}\geqslant t_1$. 证毕.

定理 2 设当前为第t代, 在至今最优信息素更新方式下, 有

$$\mathrm{LBP}(t)=[|S_{<=}|\prod_{i=1}^{n}\frac{\tau_{\min}^\alpha}{\tau_{\min}^\alpha+(|D_i|-1)\tau_{\max}^\alpha}]^{m\cdot\lceil\frac{\ln\tau_{\min}-\ln\tau_{\max}}{\ln(1-\rho)}\rceil},\tag{4}$$

其中: $S_{<=}=\{s\in S|f(s)\geqslant f(\hat{s})\}, i=1,2,\cdots,n$, $j=1,2,\cdots,|D_i|$, α为启发因子, m为蚂蚁数目.

证 由式(1)可知, 在没有约束条件的情况下, 生成一个解s的概率为

$$P_{\rm s}=\prod_{i=1}^{n}P(c_i^j|S^p)=\prod_{i=1}^{n}\frac{(\tau_i^j)^\alpha}{\sum\limits_{k=1}^{|D_i|}(\tau_i^k)^\alpha},$$

则

$$P_{\rm s}\geqslant\prod_{i=1}^{n}\frac{\tau_{\min}^{\alpha}}{\sum\limits_{k=1}^{|D_i|}(\tau_i^k)^{\alpha}}\geqslant\prod_{i=1}^{n}\frac{\tau_{\min}^{\alpha}}{\tau_{\min}^{\alpha}+(|D_i|-1)\tau_{\max}^{\alpha}}.\quad(5)$$

设c_i^l为集合$S_{<=}$中第l个元素其第i个解分量值$(i=1,2,\cdots,n,\ l=1,2,\cdots,|S_{<=}|)$, τ_i^l为其信息素值, 记$P_{\rm bs}(t)$为在第t代生成的解都不比$\hat{s}(t)$好的概率, 那么

$$P_{\rm bs}(t)=\{\sum_{l=1}^{|S_{<=}|}[\prod_{i=1}^{n}\frac{(\tau_i^l)^{\alpha}}{\sum\limits_{k=1}^{|D_i|}(\tau_i^k)^{\alpha}}]\}^m\geqslant$$

$$\{\sum_{l=1}^{|S_{<=}|}[\prod_{i=1}^{n}\frac{\tau_{\min}^{\alpha}}{\tau_{\min}^{\alpha}+(|D_i|-1)\tau_{\max}^{\alpha}}]\}^m=$$

$$\{|S_{<=}|\prod_{i=1}^{n}\frac{\tau_{\min}^{\alpha}}{\tau_{\min}^{\alpha}+(|D_i|-1)\tau_{\max}^{\alpha}}\}^m.\quad(6)$$

由于信息素的更新, 有

$$P_{\rm bs}(t+1)\geqslant$$

$$\{\prod_{i=1}^{n}[\frac{(1-\rho)\tau_i^j+g(\hat{s}(t))}{(1-\rho)\tau_i^j+g(\hat{s}(t))+(1-\rho)\sum\limits_{k=1}^{|D_i|-1}(\tau_i^k)^{\alpha}}]+$$

$$\sum_{l=1}^{|S_<|}[\prod_{i=1}^{n}\frac{(1-\rho)(\tau_i^l)^{\alpha}}{(1-\rho)\sum\limits_{k=1}^{|D_i|}(\tau_i^k)^{\alpha}+g(\hat{s}(t))}]\}^m,\quad(7)$$

式中: $c_i^j\in\hat{s}(t)$, $S_<=S_{<=}\setminus\{\hat{s}(t)\}$,

$$P_{\rm bs}(t+1)\geqslant\{\prod_{i=1}^{n}[\frac{(\tau_i^j)^{\alpha}}{(\tau_i^j)^{\alpha}+\sum\limits_{k=1}^{|D_i|-1}(\tau_i^k)^{\alpha}}]+$$

$$\sum_{l=1}^{|S_<|}[\prod_{i=1}^{n}\frac{\tau_{\min}^{\alpha}}{\tau_{\min}^{\alpha}+\sum\limits_{k=1}^{|D_i|-1}\tau_{\max}^{\alpha}}]\}^m\geqslant$$

$$\{\prod_{i=1}^{n}[\frac{\tau_{\min}^{\alpha}}{\tau_{\min}^{\alpha}+\sum\limits_{k=1}^{|D_i|-1}(\tau_{\max})^{\alpha}}]+$$

$$\sum_{l=1}^{|S_<|}[\prod_{i=1}^{n}\frac{\tau_{\min}^{\alpha}}{\tau_{\min}^{\alpha}+\sum\limits_{k=1}^{|D_i|-1}\tau_{\max}^{\alpha}}]\}^m=$$

$$[|S_{<=}|\prod_{i=1}^{n}\frac{\tau_{\min}^{\alpha}}{\tau_{\min}^{\alpha}+(|D_i-1|)\tau_{\max}^{\alpha}}]^m.\quad(8)$$

同理, 可得

$$P_{\rm bs}(t+k)\geqslant[|S_{<=}|\prod_{i=1}^{n}\frac{\tau_{\min}^{\alpha}}{\tau_{\min}^{\alpha}+(|D_i-1|)\tau_{\max}^{\alpha}}]^m.\quad(9)$$

因此, 可以得到

$$\mathrm{LPB}(t)=\prod_{i=1}^{t^c}P_{\rm bs}(t+i)\geqslant$$

$$\prod_{i=1}^{t^c}[(|S_{<=}|\prod_{i=1}^{n}\frac{\tau_{\min}^{\alpha}}{\tau_{\min}^{\alpha}+(|D_i-1|)\tau_{\max}^{\alpha}})^m].\quad(10)$$

由引理1、引理2, 得

$$\mathrm{LPB}(t)=\prod_{i=1}^{t^c}P_{\rm bs}(t+i)\geqslant$$

$$[|S_{<=}|\prod_{i=1}^{n}\frac{\tau_{\min}^{\alpha}}{\tau_{\min}^{\alpha}+(|D_i|-1)\tau_{\max}^{\alpha}}]^{m\cdot\lceil\frac{\ln\tau_{\min}-\ln\tau_{\max}}{\ln(1-\rho)}\rceil}.\quad(11)$$

显然LPB(t)的值越大, 越难提高解的质量.

4　可控搜索偏向的二元蚁群算法设计(The design of binary ant colony algorithm with controllable search bias)

根据定理2可以得出, 没有发现较优解的概率下限与$|S_{<=}|$、蚂蚁数量以及$\tau_{\min}$和$\tau_{\max}$有关. $|S_{<=}|$越大, 则搜索到较优解的概率越高. 蚂蚁数量越大, 找到较优解的概率也越大. 对于某一问题, 相对于至今最优解$\hat{s}(t)$来说, $|S_{<=}|$固定不变. $\tau_{\min}$和$\tau_{\max}$的设置可以根据文献[9], 而蚂蚁数量的增多会导致算法效率的下降.

但是在ACO中, 由于信息素的分布, 特别是在信息素落差非常大时, 蚂蚁生成的解会相同. 所以, 对于本代已生成的某个解分量值, 本文希望可以降低其被选中的概率, 以期望生成更丰富的解, 使得蚂蚁的效率达到最大.

在使用简化后的二进制网络[3]中(如图1所示), 对每个结点v_j引入访问量v_j^r, $r\in\{0,1\}$. v_j^r在每代开始时置0, 在人工蚂蚁构建解的过程中, 若v_j选择解分量值r, 则访问量v_j^r加1. 结点v_j的解分量值的选择如式(12)(13)所示:

$$\begin{cases}P_j^1=\dfrac{h(v_j^1,m)\cdot\tau_j^1(t)}{h(v_j^1,m)\cdot\tau_j^1(t)+(1-h(v_j^1,m))\cdot\tau_j^0(t)},\\ P_j^0=1-P_j^1,\ \text{如果}\ q<w_2,\end{cases}\quad(12)$$

$$\begin{cases}P_j^0=\dfrac{h(v_j^0,m)\cdot\tau_j^0(t)}{h(v_j^0,m)\cdot\tau_j^0(t)+(1-h(v_j^0,m))\cdot\tau_j^1(t)},\\ P_j^1=1-P_j^0,\ \text{如果}\ q\geqslant w_2.\end{cases}\quad(13)$$

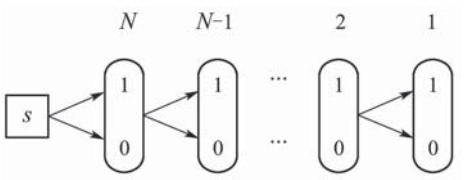

图 1　简化后的二进制网络

Fig. 1　Simplified binary network

式(12)(13)中, p_j^1, p_j^0分别是结点v_j选择解分量1和解分量0的概率, m为总蚂蚁数量, $\tau_j^r(t)$为结点v_j解分量值r的信息素, $h(v_j^r,m)=|1-w_1v_j^r/m|$, $r\in\{0,1\}$. q为$(0,1)$之间的一个随机数, $0\leqslant w_1\leqslant 2$,

$0 \leqslant w_2 \leqslant 1$. w_1, w_2为两个参数, 其作用是控制搜索偏向.

算法具体流程如下:

Step 1 初始化参数;

Step 2 蚂蚁根据式(12)(13)生成图1中每一个结点的解分量值, 并对相应的访问量加1;

Step 3 解的评价与信息素更新量的计算;

Step 4 信息素轨迹更新以及每个访问量清零;

Step 5 判断是否满足循环结束条件, 满足结束, 不满足则转至Step2;

Step 6 输出结果.

5 可控搜索偏向的二元蚁群算法分析 (Analysis of binary ant colony algorithm with controllable search bias)

首先对式(12)(13)做一下变换. 在式(12)中有

$$P_j^1=\frac{h(v_j^1,m)\cdot\tau_j^1(t)}{h(v_j^1,m)\cdot\tau_j^1(t)+(1-h(v_j^1,m))\tau_j^0(t)} \Leftrightarrow$$

$$[h(v_j^1,m)\cdot\tau_j^1(t)+(1-h(v_j^1,m))\tau_j^0(t)]P_j^1 = h(v_j^1,m)\tau_j^1(t) \Leftrightarrow$$

$$\frac{P_j^1}{1-P_j^1}=\frac{h(v_j^1,m)}{1-h(v_j^1,m)}\frac{\tau_j^1}{\tau_j^0}. \tag{14}$$

又因为$P_j^0 = 1 - P_j^1$, 所以有

$$\frac{P_j^1}{P_j^0}=\frac{h(v_j^1,m)}{1-h(v_j^1,m)}\cdot\frac{\tau_j^1}{\tau_j^0}.$$

将$h(v_j^r,m)=|1-\frac{w_1v_j^r}{m}|, r\in\{0,1\}$代入得

$$\frac{P_j^1}{P_j^0}=\frac{|1-\frac{w_1v_j^1}{m}|}{1-|1-\frac{w_1v_j^1}{m}|}\cdot\frac{\tau_j^1}{\tau_j^0}. \tag{15}$$

类似地, 可以由式(13)得出

$$\frac{P_j^0}{P_j^1}=\frac{|1-\frac{w_1v_j^0}{m}|}{1-|1-\frac{w_1v_j^0}{m}|}\cdot\frac{\tau_j^0}{\tau_j^1}. \tag{16}$$

下面具体分析.

在$h(v_j^r,m) = |1 - w_1v_j^r/m|$的函数如图2所示, 可以看到:

1) 如果$w_1\leqslant 1$, 意味着$h(v_j^r,m)$随着v_j^r的增长而减少. 那么$h(v_j^r,m)$的值从1开始递减. w_1越接近1, $h(v_j^r,m)$的值递减的越快. 如果$w_1 = 1$, $h(v_j^r,m)$以$1/m$的速率减小, 若$w_1 = 0$, 那么$h(v_j^r,m)\equiv 1$.

2) 如果$w_1 > 1$, 意味着$w_1\cdot v_j^r$可能大于m. 一旦$w_1\cdot v_j^r$大于m, $h(v_j^r,m)$从关于v_j^r的单调减函数, 变成关于v_j^r的一个单调增函数. 同样, 变化速度也由w_1决定, w_1越大, $h(v_j^r,m)$增加越快.

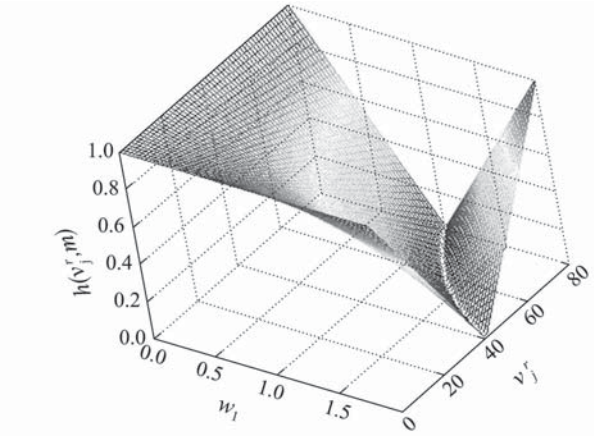

图 2 函数$h(v_j^r,m)$的图 ($m = 80$)

Fig. 2 Figure of $h(v_j^r,m)$, where $m = 80$

蚁群优化算法是MBS(model-based search)[10]算法的一种. 在MBS算法中, 新的解由参数化的概率模型所决定, 而蚁群优化概率模型的主要组成部分是信息素[11]. 在二元蚁群算法中, 不仅不设启发式信息, 而且一个结点v_j只有两个解分量值: 0和1. 所以τ_j^1与τ_j^0的比值显得尤为重要. 因此把本文算法的搜索过程大致分为3种情况: $\tau_j^1\gg\tau_j^0$, $\tau_j^1\ll\tau_j^0$, $\tau_j^1\approx\tau_j^0$.

1) $\tau_j^1\approx\tau_j^0$.

在这种情况下$\tau_j^1/\tau_j^0\approx 1$, 则式(14)(15)变为:

$$\frac{P_j^1}{P_j^0}=\frac{|1-\frac{w_1v_j^1}{m}|}{1-|1-\frac{w_1v_j^1}{m}|}, \tag{17}$$

$$\frac{P_j^0}{P_j^1}=\frac{|1-\frac{w_1v_j^0}{m}|}{1-|1-\frac{w_1v_j^0}{m}|}. \tag{18}$$

在起始时, $v_j^1=0$, $v_j^0=0$, 则式(16)推出$P_j^1/P_j^0 = \infty$, 即$P_j^1 = 1$, $P_j^0 = 0$. 若要使$P_j^1 = P_j^1 = 0.5$, 必须使$h(v_j^j,m) = |1-w_1\cdot v_j^1/m| = 0.5$, 即$w_1v_j^1 = 0.5m$或$w_1v_j^1 = 1.5m$. 在$w_1 = 1$时, 只有当$v_j^1 = 0.5m$时, 才有$P_j^1 = P_j^1 = 0.5$. 而一旦有$v_j^1 = 0.5m$, 则说明结点$v_j$选择解分量值1的蚂蚁数已经达到半数, 这也意味着, 当$w_1 = 1$时, 选择解分量值1的次数要比选择0的次要多. 显然w_1越小, 选择解分量值1的次数越多, 当$w_1 = 0$时, 则永远选择解分量值1. 当$w_1 = 2$, 则有结点v_j选择解分量值1的数量和选择0的数量大致相同. 也就是说, 式(16)会更倾向于选择解分量值1. 同理, 可以得出式(17)更倾向于选择解分量值0. w_2的大小决定蚂蚁到底倾向于使用式(16)来更多的选择解分量值1, 还是使用式(17)来更多的选择解分量值0.

2) $\tau_j^1\gg\tau_j^0$, $\tau_j^1\ll\tau_j^0$.

这两种情况基本上相同, 若$\tau_j^1\gg\tau_j^0$, 则τ_j^1/τ_j^0为一

个比较大的数, 此时对于式(14), 需要较大的v_j^1, 才能使$P_j^1 = P_j^1 = 0.5$, 也就是说, 会比$\tau_j^1 \approx \tau_j^0$这种情况更多的选择解分量值1. 而对于式(15), $P_j^1 = P_j^1$会很快的由∞向0变化, 也就是说会比$\tau_j^1 \approx \tau_j^0$这种情况更少的选择解分量值0. 反之, 如果$\tau_j^1 \ll \tau_j^0$, 情况完全类似.

由上面的分析可知, 结点v_j解分量值的选择, 不仅由τ_j^1/τ_j^0决定, 而且也由参数w_1和w_2决定, 更与访问量v_j^r有关. 总体来说: v_j^1越大, 则更倾向于选择解分量值0; v_j^0越大, 则更倾向于选择解分量值1. 参数w_1控制概率P_j^1和P_j^0从0→1或1→0变化的速度, 而参数w_2控制算法偏向于解分量值1, 还是0. 因此, 如果把算法搜索过程看作是一个雷达, 那么, w_1控制扫描的速度, w_2控制扫描的区域.

显然此信息素模型不仅能使算法趋向于信息素浓度较高的地方, 使算法产生正反馈, 保证算法的收敛性, 也能兼顾信息素浓度较低的地方, 形成负反馈, 保证了利用(exploit), 又保证了开发(explore).

6 可控搜索偏向的二元蚁群算法的复杂度分析(Analysis of complexity of binary ant colony algorithm with controllable search bias)

6.1 算法的时间复杂度分析(Analysis of time complexity)

在实际中通常采用时间复杂度的渐进法, 说明程序执行步骤的数量级, 从而估算算法执行效率的高低. 在算法的设计和分析中, 沿用实用性的复杂度概念, 即把求解时间的关键操作(如加、减、乘、除、比较等运算)指定为基本操作, 通常把算法执行基本操作的次数定义为算法的时间复杂度[12].

设m为算法蚂蚁的个数, n为二进制网络的长度, 最大循环次数为$N_{\rm c}$, C_1, C_2, C_3, C_4为常数. 根据第4节的算法流程可逐步分析出其时间复杂度.

对于Step1, 因为共有n个结点, 每个结点有两个解分量值, 对每个结点的信息素设初值, 需要$2n$次操作. 每个解分量值有1个访问量值, 故需要$2n$次操作设其初值. n结点的参数w_1, w_2初始化需要$C_1{\cdot}n$次, 其他操作需要常数次, 故Step1的时间复杂度为$2n + 2n + C_1{\cdot}n = \mathrm{O}(n)$. Step2中, m只蚂蚁需要爬过n个结点, 需要$C_2{\cdot}m{\cdot}n$次, 故时间复杂度为$C_2{\cdot}m{\cdot}n = \mathrm{O}(m{\cdot}n)$. Step3中, 对解进行评价需要$C_3$次操作, m个解需要$C_3{\cdot}m$次操作. 计算信息素更新量需要常数次操作, 故其时间复杂度为$\mathrm{O}(m)$. Step4中, 对n个结点的信息素更新, 因为每个结点有两个解分量值, 故需要$2n$次, 而访问量清零操作$2n$次, 因此Step4其时间复杂度为$2n + 2n = \mathrm{O}(n)$. Step5判断算法是否结束以及Step6输出结果其时间复杂度均为$\mathrm{O}(1)$.

Step2∼Step5需要经过$N_{\rm c}$次循环, 所以整个算法的时间复杂度为$\mathrm{O}(n) + N_{\rm c}{\cdot}\mathrm{O}(m{\cdot}n) + N_{\rm c}{\cdot}\mathrm{O}(m) + N_{\rm c}{\cdot}\mathrm{O}(n) + N_{\rm c}{\cdot}\mathrm{O}(1)$. 因此总体时间复杂度为

$$T = \mathrm{O}(N_{\rm c} \cdot m \cdot n). \tag{19}$$

本文算法额外增加的操作有: 1) 每只蚂蚁根据式(12)(13)选取下一解分量值, 需要读取参数w_1, w_2以及相应访问量值, 增加$3{\cdot}N_{\rm c}{\cdot}m{\cdot}n$次操作. 2) 当蚂蚁选取了解分量值后, 需要对相应的解分量值的访问量加1, 增加了$N_{\rm c}{\cdot}m{\cdot}n$次操作. 3) 参数的初始化, 需要$C_4{\cdot}n$次操作. 4) 访问量的清零, 需要$N_{\rm c}{\cdot}2n$次操作. 因此, 本文算法虽然增加了一些额外操作, 但总体时间复杂度仍为$\mathrm{O}(N_{\rm c}{\cdot}m{\cdot}n)$.

6.2 算法的空间复杂度分析(Analysis of space complexity)

在实际应用中, 通常把算法执行时间内所占有的存储单元定义为算法的空间复杂度[12].

算法的数据主要实现两方面功能: 1) 问题的描述. 2) 实现算法功能的辅助数据. 需要记录每个结点两个解分量值的信息素以及其访问量, 共有n个结点, 需要存储空间是$4n$. 每一个结点需要2个控制参数w_1, w_2, 需要存储空间$2n$, 为了评价解的优劣, 需要定义中间变量, 需要存储空间是$n{\cdot}m$. 渐进空间复杂定义与渐进时间复杂度的定义类似, 通过对算法各步骤的综合分析, 可得整个计算过程的空间复杂度为

$$S = \mathrm{O}(n) + \mathrm{O}(m{\cdot}n). \tag{20}$$

本文算法需要增加$2n$个空间用来存储访问量, 并且需要$2n$个存储空间存放参数w_1, w_2, 所以额外消耗了$4n$个存储空间, 因$\mathrm{O}(n) + 4n = \mathrm{O}(n)$, 额外增加的空间复杂度为$\mathrm{O}(n)$, 所以总体空间复杂度亦未变.

7 实验与结果(Experiment and results)

7.1 参数设置(Settings of parameters)

本小节参数的设置只针对w_1, w_2而言, 其他参数均与文献[3]相同.

是否存在“好的偏向”的特性是根据问题的不同而不同的[11]. 因此, 对于本文算法参数w_1, w_2的设置分为两种情况: 1) 已知“好的偏向”的情况. 2) 未知“好的偏向”的情况.

因为问题的不同, 本文对情况1)无法给出通用的方法, 只在第7.3节结合多背包问题, 给出一个具体的应用实例. 对于情况2), 本文给出两种不同的方法, 以做参考:

1) 平衡搜索偏向的参数取法. w_1, w_2一律取2,

0.5. 由第5节可知, w_1越小, P_j^1和P_j^0从0→1或1→0变化的速度越慢, $w_1 = 0, w_2 = 0.5$时, 算法变为纯随机算法. 当算法在搜索过程中过多的产生某一解分量值时, 本文希望可以降低其产生的概率, 而概率的变化速度由w_1调控, 所以把w_1设为2, 可以将这种变化速度达到最大, 以尽量平衡解分量值之间的搜索偏向. 同时出于同样的目的, 将w_2设为0.5, 使式(12)(13)有着均等的概率. 在这里将w_2设为2是由于$h(v_j^r, m)$的限制, 若将$h(v_j^r, m) = |\cos(w_1 v_j^r/m)|$, 此时$w_1$的取值没有任何限制, 但$w_1$不宜过大, 否则影响算法的收敛.

2) 随机抽样的参数取法. 此方法仅适用于实时性低且计算评价函数时间花费少的问题. 结点v_j $(j = 1, 2, \cdots, n)$的参数w_2记为w_{2j}. 为了使式(12)(13)带有偏向, w_1应取较小的值, 但由于种群多样性的必要性, w_1的取值又不宜太小, 结合图2以及实验结果, w_1取0.6∼1.0较佳. 对于w_2的取值, 首先对搜索空间进行随机取Y(如10^6)个点, 以评估空间中的偏向, 包含$v_j = 1$的点的集合记为C_j^1, $\bar{C}_j^1$为其适应度均值, 包含$v_j = 0$的点的集合记为C_j^0, 其适应度均值记为$\bar{C}_j^0$. θ_j^0为C_j^0中适应度最大的前$\lceil \alpha \cdot C_j^0 \rceil$个点的集合, 其中$\alpha \in (0, 1)$是常数, 表示占有$|C_j^0|$的百分比, θ_j^1的定义类似. 有

$$\bar{\theta}_j^0 = (\sum_{i=1}^{|\theta_j^0|} y_i)/|\theta_j^0|, \tag{21}$$

其中: $y_i = f(x_i^d)$, 点$\{x_i^d\} \in \theta_j^0$, $f(\cdot)$为适应度函数, d为搜索空间的维数. 同样的,

$$\bar{\theta}_j^1 = (\sum_{i=1}^{|\theta_j^1|} y_i)/|\theta_j^1|. \tag{22}$$

定义w_{2j}为

$$w_{2j} = \frac{\bar{C}_j^1 \cdot \bar{\theta}_j^1}{\bar{C}_j^1 \cdot \bar{\theta}_j^1 + \bar{C}_j^0 \cdot \bar{\theta}_j^0}. \tag{23}$$

7.2 函数优化问题(Function optimization problem)

本文所用的测试函数请见附录, 表1给出了本文所要比较的算法. 当前, 大多数关于函数优化问题的论文使用目标函数的评价次数作为评测标准, 本文也不例外. 使用这种评测方法的好处是: 消除了由于编程技巧、不同编程语言以及机器原因带来的差异[13].

表 1 本文所比较的算法

Table 1 The compared algorithms

算法	引自	算法	引自
ACO_R	文献[13]	CIAC	文献[16]
ACORSES	文献[14]	API	文献[17]
MACO	文献[15]	CACO	文献[18]

由于随机抽样的参数取法需要额外目标函数评价次数, 因此将与其他算法分开讨论. 表2、表3给出了各算法所需的平均评价次数, 参数的取法为平衡搜索偏向的方法. 表2、表3中评价次数均是独立运行50次后的平均值, 停机准则为

$$|f - f^*| < \varepsilon_1 f + \varepsilon_2,$$

其中: f^*为最优解, $\varepsilon_1 = \varepsilon_2 = 10^{-4}$.

表 2 各算法所需的平均评价次数

Table 2 Average numbers of function evaluations

函数	本文算法	ACO_R	CIAC	API	CACO
R_2	416	820	11480	9840	6806
R_5	2376	2487	39792	—	—
SM	683	781	49984	10153	21868
GP	110	392	23424	—	—
B_2	136	559	11968	—	—
ES	738	772	—	—	—
DJ	142	392	—	—	—

表 3 各算法所需的平均评价次数

Table 3 Average numbers of function evaluation

测试函数	算法	最优解	评价次数
R_2	MACO	0	3600
	ACORSES	0	2174
	本文算法	0	1012
f_{no1}	MACO	−10	3750
	ACORSES	−10	2167
	本文算法	−10	992
f_{no2}	MACO	—	—
	ACORSES	−837.9658	1176
	本文算法	−837.9658	1198
f_{no3}	MACO	−2	235
	ACORSES	−2	1610
	本文算法	−2	512
f_{no4}	MACO	1	36000
	ACORSES	1	1576
	本文算法	1	1232
f_{dj}	MACO	—	—
	ACORSES	0	4850
	本文算法	0	488

表2中, 本文算法在所有测试函数的表现均优于其他算法, 在某些测试函数上的评价次数甚至只有其他算法的几十分之一. 在表3中, 本文算法在f_{no2}上的表现略差于ACORSES, 以及f_{no3}上逊于MACO, 除此之外, 全部优于其他算法, 体现了本文算法良好的性能以及鲁棒性.

图3横坐标表示迭代次数, 纵坐标表示目标函数值, 给出了本文算法在测试函数R_2上两种不同参数取法的比较(随机抽样取法和平衡搜索偏向的取法), 抽样次数为10^6. 在不考虑抽样所花费的评价次数的情况下, 随机抽样参数取法在求解精度以及收敛速度上均优于平衡搜索偏向的参数取法, 但随机抽样参数取法随着对搜索空间进行抽样, 需要额外计算大量的目标函数值, 适用性不强.

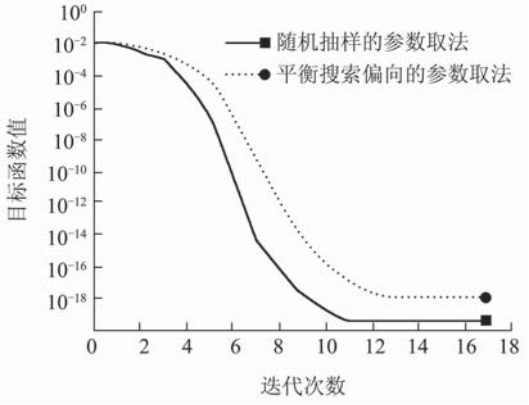

图 3 本文算法两种参数取法的性能比较

Fig. 3 Comparison between two different parameter settings

7.3 多背包问题(Multiple knapsack problem)

多背包问题(Multiple knapsack problem, MKP)是经典的组合优化问题, 该问题的描述如下: 存在价值分别为$Va[1], Va[2], \cdots, Va[n]$的$n$个物品和编号从1到$m$的$m$个背包, 其容积分别为$C[1], C[2], \cdots, C[m]$; 每个物品装在不同的背包中的体积是不同的, 因此存在以下体积矩阵:

$$VO = \begin{pmatrix} Vo_{1,1} & Vo_{1,2} \cdots & Vo_{1,n} \\ Vo_{2,1} & Vo_{2,2} \cdots & Vo_{2,n} \\ \vdots & \vdots & \vdots \\ Vo_{m,1} & Vo_{m,2} \cdots & Vo_{m,n} \end{pmatrix}, \quad (24)$$

其中$Vo_{i,j}$表示j号物品装入i号背包时所占用的体积, $Vo_{i,j}=0$表示j号物品不允许放入i号背包. 在保证每个背包中物品的体积之和不超过各背包的容积的前提下, 要求被装所有物品的总价值最大化:

$$\max \sum_{i=1}^{m}\sum_{j=1}^{n} x_{i,j}\cdot Va[j],$$

$$\text{s.t.} \begin{cases} \forall j\in\{1,2,\cdots,n\},\ \sum\limits_{i=1}^{m} x_{i,j}\leqslant 1; \\ \forall i\in\{1,2,\cdots,m\},\ \sum\limits_{j=1}^{n} x_{i,j}\cdot Vo_{i,j}j\leqslant C[i]; \\ \forall i\in\{1,2,\cdots,m\},\ \forall j\in\{1,2,\cdots,n\}, \\ Vo_{i,j}-x_{i,j}>-1, \end{cases} \quad (25)$$

其中$x_{i,j}\in\{0,1\}$, $x_{i,j}=1$表示j号物品装入i号背包, $x_{i,j}$的$m\times n$的矩阵X即为问题的解.

7.3.1 多背包问题的参数设置(Parameters settings in multiple knapsack problem problem)

对于多背包问题来说, 解的质量与装入背包物品的价值正相关, 物品价值越大, 解的质量越高, 显然此为一“好的偏向”. 文献[11]中指出, 物品的体积与其被高质量解所包含的次数成强烈的负相关, 也就是说, 平均质量越高的解集, 其所包含体积越大的物品的次数越小, 此为另一“好的偏向”. 因此, 本文将$Va[j]/Vo_{i,j}$作为参数设置的依据.

借用模糊数学中的贴近度[19]来确定参数w_2. 贴近度定义如下:

$$N_{\rm m}(A,B)=\frac{\sum\limits_{i=1}^{n} A(x_i)\wedge B(x_i)}{\sum\limits_{i=1}^{n} A(x_i)\vee B(x_i)}, \quad (26)$$

其中: $A(x_i)$和$B(x_i)$分别表示A, B的元素, $\wedge$, $\vee$分别表示取小和取大运算符.

记$Vao_{i,j}=Va[j]/Vo_{i,j}$, $\overline{Vao}_{i,j}(j=1,\cdots,n)$为其均值, $N_{{\rm m}(i,j)}=N_{\rm m}(Vao_{i,j},\overline{Vao}_{i,j})$, 那么定义$w_{2ij}$为

$$w_{2ij}=\begin{cases} 0.5, & Vao_{i,j}=\overline{Vao}_{i,j}, \\ 0.5\cdot N_{{\rm m}(i,j)}, & Vao_{i,j}<\overline{Vao}_{i,j}, \\ 1-0.5N_{{\rm m}(i,j)}, & Vao_{i,j}>\overline{Vao}_{i,j}, \end{cases} \quad (27)$$

其中: $i=1,\cdots,m$, $j=1,\cdots,n$, 对于不允许放入的位置则不处理. w_1取值与随机抽样参数取法一致, 以下均取$w_1=0.75$.

7.3.2 多背包问题实验结果(Results of multiple knapsack problem)

本文多背包问题所采取的测试数据均来自http://elib.zib.de/pub/Packages/mp-testdata/ip/sac94-suite/index.html上记载的经典测试集.

测试数据 1　pb4.

该问题的规模为$m=2$, $n=29$. 物品的价值矩阵为:

$$\begin{aligned} Va=(&7074,5587,5500,3450,367,4235,9262,\\ &6155,32240,1600,4500,6570,7010,16020,\\ &8000,2568,2365,4350,4975,29400,7471,\\ &3840,3575,1125,1790,2450,540,445,220). \end{aligned}$$

体积矩阵$Vo=[B_1\ \ B_2]^{\rm T}$, 其中0表示不允许放入.

$$\begin{aligned} B_1=(&25,17,20,22,15,10,50,10,150,0,0,0,40,\\ &60,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0), \end{aligned}$$

$$\begin{aligned} B_2=(&0,0,0,0,0,2,5,6,40,2,6,10,13,30,15,5,\\ &5,10,15,91,24,15,15,5,10,15,10,10,10). \end{aligned}$$

背包容积为$C=(153,154)^{\mathrm{T}}$. 根据式(26)确定参数矩阵$w_2=[w^{21}\ \ w^{22}]^{\mathrm{T}}$, 其中:

$$\begin{aligned}w^{21}=(&0.18,0.85,0.82,0.68,0.12,0.88,0.73,\\&0.92,0.76,0,0,0,0.71,0.81,0,0,0,0,\\&0,0,0,0,0,0,0,0,0,0,0),\\w^{22}=(&0,0,0,0,0,0.89,0.88,0.78,0.72,0.72,\\&0.70,0.65,0.58,0.57,0.57,0.56,0.52,\\&0.48,0.36,0.36,0.34,0.28,0.26,\\&0.25,0.20,0.18,0.06,0.05,0.02).\end{aligned}$$

实验结果如表4所示.

表 4 pb4的结果

Table 4 Results of pb4

已知最优解	最优解	均值	方差	评价次数
95168	136567	136567	0	772.8

其中一个解的矩阵X如下:

$$X=\begin{pmatrix}1&1&1&1&0&0&0&0&0&0&0&0&0&0&1&0\\0&0&0&0&0&1&1&1&1&1&1&1&1&1&0&1\\0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\1&0&1&1&0&1&0&0&0&0&0&0&0&0&0&0\end{pmatrix}.$$

采用平衡搜索偏向的参数取法, 其结果如表5所示.

表 5 pb4平衡搜索偏向参数取法的结果

Table 5 Results of pb4 of balanced search bias

已知最优解	最优解	均值	方差	评价次数
95168	136567	136561	16.49	896

测试数据 2 weing2.

该问题的规模为$m=2,n=28$. 物品的价值矩阵为:

$$\begin{aligned}Va=(&1898,440,22507,270,14148,3100,4650,\\&30800,615,4975,1160,4225,510,11880,\\&479,440,490,330,110,560,24355,2885,\\&11748,4550,750,3720,1950,10500).\end{aligned}$$

体积矩阵$Vo=[B_1\ \ B_2]^{\mathrm{T}}$, 其中0表示不允许放入.

$$\begin{aligned}B_1=(&45,0,85,150,65,95,30,0,170,0,40,25,20,\\&0,0,25,0,0,25,0,165,0,85,0,0,0,0,100),\\B_2=(&30,20,125,5,80,25,5,73,12,15,15,40,5,\\&10,10,12,10,9,0,20,60,40,50,36,49,\\&40,19,150).\end{aligned}$$

背包容积为$C=(500,500)^{\mathrm{T}}$, 参数矩阵$w_2=[w^{21}\ \ w^{22}]^{\mathrm{T}}$, 其中:

$$\begin{aligned}w^{21}=(&0.44,0,0.91,0.02,0.89,0.34,0.84,0,\\&0.04,0,0.3,0.86,0.26,0,0,0.18,0,0,0.05,\\&0,0.84,0,0.83,0,0,0,0,0.77),\\w^{22}=(&0.17,0.06,0.49,0.15,0.48,0.34,0.90,\\&0.78,0.14,0.72,0.21,0.29,0.28,0.92,0.13,\\&0.10,0.13,0.10,0.00,0.08,0.77,0.20,0.61,\\&0.34,0.04,0.25,0.28,0.19).\end{aligned}$$

实验结果如表6所示.

表 6 weing2的结果

Table 6 Results of weing2

已知最优解	最优解	均值	方差	评价次数
130883	164045	164045	0	1770.4

其中一个解的矩阵X如下:

$$X=\begin{pmatrix}1&0&1&0&1&1&1&0&0&0&0&1&0&0\\0&1&0&1&0&0&0&1&1&1&1&0&1&1\\0&1&0&0&1&0&0&0&0&0&0&0&0&1\\1&0&1&1&0&1&1&1&1&1&1&1&1&0\end{pmatrix}.$$

采用平衡搜索偏向的参数取法, 其结果如表7所示.

表 7 weing2平衡搜索偏向参数取法的结果

Table 7 Results of weing2 of balanced search bias

已知最优解	最优解	均值	方差	评价次数
130883	164045	164020	20	1790.4

测试数据 3 weish01.

该问题的规模为$m=5,n=30$. 物品的价值矩阵为:

$$\begin{aligned}Va=(&360,83,59,130,431,67,230,52,93,125,\\&670,892,600,38,48,147,78,256,63,17,120,\\&164,432,35,92,110,22,42,50,323).\end{aligned}$$

体积矩阵为$Vo=[B_1\ \ B_2\ \ B_3\ \ B_4\ \ B_5]^{\mathrm{T}}$, 其中0表示不能放入.

$$\begin{aligned}B_1=(&7,0,30,22,80,94,11,81,70,64,59,18,0,\\&36,3,8,15,42,9,0,42,47,52,32,26,48,\\&55,6,29,84),\\B_2=(&8,66,98,50,0,30,0,88,15,37,26,72,61,\\&57,17,27,83,3,9,66,97,42,2,44,71,11,\\&25,74,90,20),\\B_3=(&3,74,88,50,55,19,0,6,30,62,17,81,25,\end{aligned}$$

$46, 67, 28, 36, 8, 1, 52, 19, 37, 27, 62, 39, 84, 16, 14, 21, 5)$,

$B_4 = (21, 40, 0, 6, 82, 91, 43, 30, 62, 91, 10, 41, 12, 4, 80, 77, 98, 50, 78, 35, 7, 1, 96, 67, 85, 4, 23, 38, 2, 57)$,

$B_5 = (94, 86, 80, 92, 31, 17, 65, 51, 46, 66, 44, 3, 26, 0, 39, 20, 11, 6, 55, 70, 11, 75, 82, 35, 47, 99, 5, 14, 23, 38)$.

背包容积为$C = (400, 500, 500, 600, 600)^{\mathrm{T}}$,

$w_2 = [w^{21}\ \ w^{22}\ \ w^{23}\ \ w^{24}\ \ w^{25}]^{\mathrm{T}}$,

$w^{21} = (0.92, 0, 0.12, 0.37, 0.34, 0.04, 0.81, 0.04, 0.08, 0.12, 0.65, 0.92, 0, 0.07, 0.75, 0.78, 0.33, 0.38, 0.44, 0, 0.18, 0.22, 0.52, 0.07, 0.22, 0.14, 0.03, 0.44, 0.11, 0.24)$,

$w^{22} = (0.80, 0.04, 0.02, 0.07, 0, 0.06, 0, 0.02, 0.17, 0.09, 0.65, 0.35, 0.27, 0.02, 0.08, 0.15, 0.03, 0.90, 0.20, 0.01, 0.03, 0.11, 0.96, 0.02, 0.04, 0.28, 0.02, 0.02, 0.02, 0.45)$,

$w^{23} = (0.95, 0.04, 0.03, 0.10, 0.31, 0.14, 0, 0.34, 0.12, 0.08, 0.84, 0.43, 0.73, 0.03, 0.03, 0.20, 0.08, 0.80, 0.90, 0.01, 0.25, 0.17, 0.60, 0.02, 0.09, 0.05, 0.05, 0.12, 0.09, 0.90)$,

$w^{24} = (0.52, 0.06, 0, 0.62, 0.16, 0.02, 0.16, 0.05, 0.05, 0.04, 0.88, 0.62, 0.84, 0.29, 0.02, 0.06, 0.02, 0.16, 0.02, 0.01, 0.52, 0.95, 0.14, 0.02, 0.03, 0.70, 0.03, 0.03, 0.67, 0.17)$,

$w^{25} = (0.31, 0.08, 0.06, 0.11, 0.77, 0.69, 0.28, 0.08, 0.16, 0.15, 0.79, 0.99, 0.86, 0, 0.10, 0.57, 0.56, 0.93, 0.09, 0.02, 0.71, 0.17, 0.42, 0.08, 0.16, 0.09, 0.35, 0.24, 0.17, 0.63)$.

实验结果如表8所示.

表 8 Weish01的结果

Table 8 Results of Weish01

已知最优解	最优解	均值	方差	评价次数
4554	5829	5829	0	182.7

其中一个解的矩阵X如下:

$$X = \left\{ \begin{array}{cccccccccccccccccccccccccccccccccccc} 0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&1&0&1&0&0&0&1&0&0&1&0&0&0&1&0&0 \\ 0&0&0&1&0&0&0&0&0&1&0&0&1&0&0&1&0&1&0&0&0&1&1&0&0&0&0&0&0&0&0 \\ 1&0&0&0&1&0&0&1&1&0&1&0&0&1&0&0&0&0&1&1&0&0&0&0&1&0&0&0&0&1 \\ 0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&1&0 \\ 0&0&0&0&0&1&1&0&1&0&0&1&0&0&0&0&1&0&0&0&0&0&0&1&0&0&0&0&0&0 \end{array} \right\}.$$

采用平衡搜索偏向的参数取法, 其结果如表9所示.

表 9 Weish01平衡搜索偏向参数取法的结果

Table 9 Results of Weish01 of balanced search bias

已知最优解	最优解	均值	方差	评价次数
4554	5829	5829	0	353.8

由多背包问题的实验表明, 本文算法的求解结果均远远超过网上已公布的最优, 且求解速度快、鲁棒性强.

8 结束语(Conclusion)

在本文中, 首先得到了ACO算法在至今最优信息素更新方式下的LPB(t). 通过对LPB(t)的分析, 设计了可控搜索偏向的二元蚁群法. 该算法不仅继承了二元蚁群算法的优点, 而且提供了可控搜索偏向的算法框架.

蚁群算法寻优需要由信息素引导, 但信息素的分布也是导致算法局部收敛的根本原因. 访问量是信息素分布的一种体现, 因此将访问量与蚂蚁数目的关系作为控制搜索偏向的依据, 是合理的. 本文提供了在两种未知偏向的情况下参数w_1, w_2的取值方法, 以及结合多背包问题, 举例说明在已知“好偏向”的情况下参数的取值方法, 实验结果亦表明了算法良好的性能. 而且函数$h(v_j^r, m)$亦不仅仅限于$h(v_j^r, m) = |1 - (w_1 \cdot v_j^r / m)|$, 根据问题的不同, 可以对函数$h(v_j^r, m)$做出变化, 以适应不同实际情况.

参考文献(References):

[1] DORIGO M, MANIEZZO V, COLORNI A. 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.

[2] DORIGO M, CARO GD. Ant colony optimization: a new meta-heuristic[C] //*Proceedings of the Congress on Evolutionary Computation*. Piscataway, NJ, USA: IEEE, 1999: 1470 – 1477.

[3] 熊伟清, 魏平. 二进制蚁群进化算法[J]. 自动化学报, 2007, 33(3): 259 – 264.
(XIONG Weiqing, WEI Ping. Ant colony evolutionary algorithm[J]. *Acta Automatica Sinica*, 2007, 33(3): 259 – 264.)

[4] 严彬, 熊伟清, 程美瑛, 等. 基于拥塞控制的多种群二元蚁群算法[J]. 控制理论与应用, 2009, 26(4): 387 – 394.

(YAN Bin, XIONG Weiqing, CHEN Meiying, et al. Multi-population binary ant colony algorithm with congestion control strategy[J]. *Control Theory & Applications*, 2009, 26(4): 387 – 394.)

[5] BLUM C, DORIGO M. Deception in ant colony optimization[C] //*The 4th International Workshop on Ant Colony Optimization and Swarm Intelligence*. Berlin: Springer-Verlag, 2004, 3172: 119 – 130.

[6] BLUM C, DORIGO M. Search bias in ant colony optimization: on the role of competition-balanced systems[J]. *IEEE Transactions on Evolutionary Computation*, 2005, 9(2): 159 – 174.

[7] WALTER J, GUTJAHR. ACO algorithms with guaranteed convergence to the optimal solution[J]. *Information Processing Letters*, 2002, 82(3): 145 – 153.

[8] STüTZLE T G, DORIGO M. A short convergence proof for a class of ant colony optimization algorithms[J]. *IEEE Transactions on Evolutionary Computation*, 2002, 6(4): 358 – 365.

[9] STüTZLE T G, HOOS H H. Max-min ant system[J]. *Future Generation Computer Systems*, 2000, 16(8): 889 – 914.

[10] ZLOCHIN M, BIRATTARI M, MEULEAU N, et al. Model-based search for combinatorial optimization: a critical survey[J]. *Annals of Operations Research*, 2004, 131(1/4): 373 – 395.

[11] MONTGOMERY J, RANDALL M, HENDTLASS T. Solution bias in ant colony optimisation: Lessons for selecting pheromone models[J]. *Computers and Operations Research*, 2008, 35(9): 2728 – 2749.

[12] MANBER U. *Introduction to Algorithms: A Creative Approach*[M]. Milano, Italy: Addison-Wesley, 1989.

[13] KRZYSZTOF S, DORIGO M. Ant colony optimization for continuous domains[J]. *European Journal of Operational Research*, 2008, 185(3): 1155 – 117.

[14] BASKAN O, HALDENBILEN S, CEYLAN H, et al. A new solution algorithm for improving performance of ant colony optimization[J]. *Applied Mathematics and Computation*, 2009, 211(1): 75 – 84.

[15] TOKSAR M D. A heuristic approach to find the global optimum of function[J]. *Journal of Computational and Applied Mathematics*, 2007, 209(2): 160 – 66.

[16] DREO J, SIARRY P. Continuous interacting ant colony algorithm based on dense hierarchy[J]. *Future Generation Computer Systems*, 2004, 20(5): 841 – 856.

[17] MONMARCHE N, VENTURINI G, SLIMANE M. On how Pachycondyla apicalis ants suggest a new search algorithm[J]. *Future Generation Computer Systems*, 2000, 16(8): 937 – 946.

[18] BILCHEV G, PARMEE I C. The ant colony metaphor for searching continuous design spaces[C] //*Proceedings of the AISB Workshop on Evolutionary Computation*. Berlin, Germany: Springer-Verlag, 1995: 25 – 39.

[19] 杨纶标, 高英仪. 模糊数学原理及应用[M]. 广州: 华南理工大学出版社, 2001.
(YANG Lunbiao, GAO Yingyi. *Fuzzy Mathematics Theory and Its Application*[M]. Guangzhou: South China University of Technology Press, 2001.)

附录 本文所采用的测试函数(Appendix Test problems in this paper)

表 10 本文所采用的测试函数

Table 10 Test problems in this paper

测试函数	公式	理论最优解
Rosenbrock(R_n)	$f(\vec{x}) = 100(x_1^2 - x_2)^2 + (1 - x_1)^2,\ \vec{x}\in[-5, 10]^n,\ n = 2, 5$	$f(1, 1, \cdots, 1) = 0$
Goldstein and Price(GP), $\vec{x}\in[2, 2]^n$	$f(\vec{x}) = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)]\cdot [30 + (2x_1 + 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$	$f(0, -1) = 3$
Sphere Model(SM)	$f(\vec{x}) = \sum\limits_{i=1}^{n} x_i^2,\ \vec{x}\in[-3, 7]^n,\ n = 6$	$f(0, 0, \cdots, 0) = 0$
Bohachevsky 2(B_2)	$f(\vec{x}) = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1) - 0.4\cos(4\pi x_x) + 0.7,\ \vec{x}\in[-100, 100]^n$	$f(0, 0) = 0$
Easom(ES)	$f(\vec{x}) = -\cos(x_1)\cos(x_2)\mathrm{e}^{-(x_1-\pi)^2-(x_2-\pi)^2},\ \vec{x}\in[-100, 100]^n$	$f(\pi, \pi) = 1$
De Jong(DJ)	$f(\vec{x}) = x_1^2 + x_2^2 + x_3^2,\ \vec{x}\in[-100, 100]^n$	$f(0, 0, 0) = 1$
f_{no1}	$f(\vec{x}) = x_1/(1 + \lvert x_2\rvert),\ \vec{x}\in[-10, 10]^n$	$f(-10, 0) = -10$
f_{no2}, $\vec{x}\in[-500, 500]^n$, $n = 2$	$f(\vec{x}) = \sum_{i=1}^{n} -x\cdot\sin(\sqrt{\lvert x_i\rvert})\vec{x}\in[-500, 500]^n$	$f(420.9687^n) = -418.9829n$
f_{no3}	$f(\vec{x}) = x_1^2 + x_2^2 - \cos(18x_1) - \cos(18x_2),\ \vec{x}\in[-2, 2]^n$	$f(0, 0) = -2$
f_{no4}, $\vec{x}\in[-5, 5]^n$	$f(\vec{x}) = \mathrm{e}^{(0.5(x_1^2+x_2^2-25)^2)} + \sin^4(4x_1 - 3x_2) + 0.5(2x_1 + x_2 - 10)^2$	$f(3, 4) = 1$

作者简介:

胡 钢 (1982—), 男, 硕士研究生, 目前研究方向为自然计算, E-mail: hugangdj@tom.com;

熊伟清 (1966—), 男, 教授, 目前研究方向为自然计算、人工智能; E-mail: xiongweiqing@nbu.edu.cn;

张 翔 (1984—), 男, 硕士研究生, 目前研究方向为复杂系统;

袁军良 (1982—), 男, 硕士研究生, 目前研究方向为自然计算.