文章编号: 1001-0920 (2016) 06-0990-07 DOI: 10.13195/j.kzyjc.2015.0595

量子衍生涡流搜索算法

李盼池, 卢爱平

(东北石油大学 计算机与信息技术学院，黑龙江 大庆 163318)

摘 要: 涡流搜索是最近提出的新型优化算法, 具有操作简单且搜索能力强的突出优点, 但在后期容易陷入早熟收敛. 对比, 通过在该算法中引入量子计算, 提出一种量子衍生涡流搜索算法. 首先将涡流中心用量子比特编码; 然后将其在 Bloch 球面上实施多次旋转得到多个个体, 将最优个体作为新的涡流中心, 完成一次迭代. 对新的涡流中心再次实施旋转, 直至满足终止条件. 标准函数极值优化的实验结果表明, 所提出的方法明显优于普通涡流搜索算法.

关键词: 涡流搜索；量子比特编码；量子比特旋转；量子衍生涡流搜索

中图分类号: TP183 **文献标志码:** A

Quantum-inspired vortex search algorithm

LI Pan-chi, *LU Ai-ping*

(School of Computer and Information Technology，Northeast Petroleum University，Daqing 163318，China. Correspondent：LI Pan-chi，E-mail：lipanchi@vip.sina.com)

Abstract: The vortex search is a new optimization algorithm recently proposed, which has the advantages of simple operation and strong search capabilities. However, it is easy to fall into premature convergence in the late stages of the algorithm. By introducing quantum computing into the algorithm, a quantum-inspired vortex search algorithm is proposed. Firstly, the vortex center is encoded by qubits described on the Bloch sphere, and then through repeatedly rotating all qubits on this individual about the same coordinate axis, some new individuals are generated. The best individual is chosen as a new vortex center which is rotated again until meeting the termination conditions. The experimental results of some benchmark functions extreme optimization show that the proposed algorithm is obviously superior to the original one.

Keywords: vortex searching；qubits encoding；qubits rotating；quantum-inspired vortex searching

0 引 言

对于实际生活中的很多优化问题, 由于问题自身的一些性质 (如高维、多峰、不可微分), 解析方法往往不能得到精确解[1]. 因此, 启发式方法就成为解决这些问题的有效途径. 启发式算法能在可接受的时间内得到近似解, 但不能保证解的最优性. 启发式算法分为单解启发和群解启发两种类型[2]. 单解启发也称为轨道方法, 在任何时刻都基于单个解, 例如模拟退火[3-4]、禁忌搜索[5]、迭代局部搜索[6]、导向局部搜索[7]、变邻域搜索[8]. 群解启发在开始时建立一个种群, 然后逐代更新直到满足终止条件. 群解启发式又可分为两种: 进化算法和群集算法. 进化算法基于自然选择的概念, 根据适应度从种群中选出优良个体, 然后采用一些特定算子产生后代. 差分进化[9-10]和遗传算法[11]是进化算法的突出代表. 群集算法是另一类受动物 (如蚂蚁、蜜蜂、鱼、鸟等) 的群集行为启发的优化算法. 该算法的特点是成员都是简单个体, 通过协作在决策空间移动寻优. 蚁群优化[12]、粒子群优化[13]、人工蜂群优化[14-15]是群集算法的突出代表. 一个启发式优化算法能否获得全局最优解, 关键取决于是否有能力获得探索和开发之间的平衡. 一般说来, 单解算法偏重开发, 而群解算法则偏重探索.

受涡流模式的启发, 文献 [16] 提出了一种基于单解寻优的涡流搜索算法 (VS), 该算法采用一个新的自适应步长调整方案来调整探索开发两阶段的平衡. 文献 [16] 给出的实验结果表明, 涡流算法的搜索能力超过了单解的模拟退火算法、模式搜索算法和群解的人工蜂群算法, 而与一个改进版本的粒子群算法相当.

收稿日期: 2015-05-11；**修回日期:** 2015-08-01.

基金项目: 国家自然科学基金项目(61170132)；黑龙江省自然科学基金项目(F2015021)；黑龙江省教育厅科学技术研究项目(12541059).

作者简介: 李盼池(1969−), 男, 教授, 博士生导师, 从事量子衍生智能优化和量子衍生神经网络等研究；卢爱平(1977−), 女, 副教授, 博士生, 从事智能优化算法及在油气地质勘探中的应用研究.

尽管涡流方法有上述优点, 但在算法后期容易陷入早熟收敛. 本文的目的在于, 在文献 [16] 提出的涡流算法中引入量子计算模式, 提出一种量子衍生涡流搜索算法 (QIVS). 该方法采用量子比特对涡流中心编码, 采用在 Bloch 球面上使量子比特绕着坐标轴旋转的方法生成候选解. 在文献 [16] 中, 搜索过程通过使涡流中心的每一维在数轴上左右移动实现, 最优解的每一维均为数轴上的一个点. 而在本文提出的方法中, 通过使涡流中心的量子比特在 Bloch 球面上绕坐标轴旋转实现搜索, 最优解的每一维均为 Bloch 球面上的一个圆周. 因此, 本文提出的算法有助于增强个体多样性, 进而有助于避免在算法后期陷入早熟收敛. 实验结果验证了所提出算法的优良性能.

1　涡流搜索算法

在单解算法中, 对当前解施加小的改变时, 搜索体现出较强的局部性; 相反, 若对当前解施加大的改变, 则相当于在搜索空间随机搜索. 在初始阶段, 往往需要一个高效的探索, 而一旦算法收敛到局部解附近, 则需要进一步的局部开发, 以使当前解向着最优解逼近. 涡流算法较好地体现了这种思想.

1.1　产生初始解

在 D 维空间中, 涡流搜索可由很多嵌套的环来建模. 首先, 最外面的环定位在搜索空间的中心, 最初的中心 μ_0 可用下式计算:

$$\mu_0 = \frac{\text{upperlimit} + \text{lowerlimit}}{2}, \tag{1}$$

其中 upperlimit 和 lowerlimit 分别是 D 维向量, 表示搜索空间的边界.

1.2　产生候选解

在涡流算法中, 候选解 $C_0(s) = \{s_1, s_2, \cdots, s_n\}$ 通过以 μ_0 为中心的高斯分布随机产生. 高斯分布的一般形式如下所示:

$$p(x|\mu, C) = \frac{1}{\sqrt{(2\pi)^D C}} \exp\left\{-\frac{1}{2}(x-\mu)^{\mathrm{T}} C^{-1}(x-\mu)\right\}. \tag{2}$$

其中: x 是 $D \times 1$ 维随机变量, μ 是 $D \times 1$ 维样本均值 (涡流中心) 向量, C 是协方差矩阵.

涡流搜索采用球形高斯分布产生候选解, 此时协方差矩阵 C 按下式计算:

$$C = \sigma^2 I_{D\times D}, \tag{3}$$

其中 σ^2 为方差. 初始标准差的计算式为

$$\sigma_0 = \frac{\max(\text{upperlimit}) - \min(\text{lowerlimit})}{2}, \tag{4}$$

其中 σ_0 可看作涡流外环的初始半径, 为随后的优化过程提供了鸟瞰.

1.3　当前解的更新

用最好的候选解 $s' \in C_0(s)$ 替换当前解, 并将其作为半径 σ_1 的内环中心, 产生新的候选解集 $C_1(s)$, 若最好解 $s' \in C_1(s)$ 好于迄今为止的最好解, 则更新最好解. 再将最好解作为缩减半径后的第 3 个环中心, 重复上述过程直至满足终止条件, 如图 1 所示.

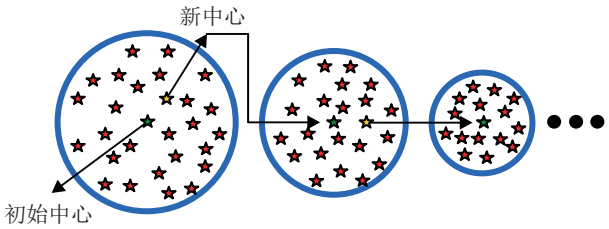

图 1　涡流算法的搜索过程

1.4　半径缩减方法

在涡流算法中, 每一步迭代采用下式不完全伽马函数的逆函数缩减半径的值:

$$\gamma(\lambda, a) = \int_0^{\lambda} \mathrm{e}^{-t} t^{a-1} \mathrm{d}t. \tag{5}$$

其中: $\lambda > 0$ 为随机变量; $a > 0$ 为分辨率参数, 在搜索过程中逐代更新为

$$a_t = a_0 - \frac{t}{\text{MaxItr}}. \tag{6}$$

为确保在开始阶段覆盖所有搜索空间, 选择 $a_0 = 1$, t 为当前步数, MaxItr 为限定步数. 每步迭代涡流半径按下式更新:

$$\sigma_t = \sigma_0 (1/\lambda)\gamma(\lambda, a_t). \tag{7}$$

文献 [16] 指出, 当 $\lambda = 0.1$ 时, 搜索性能最好.

这种更新方法可使搜索半径在前半部分线性缩小, 侧重于全局探索, 后半部分指数缩小, 侧重于局部开发, 从而可较好地实现探索与开发之间的平衡.

2　量子衍生涡流算法

2.1　量子比特的球面描述

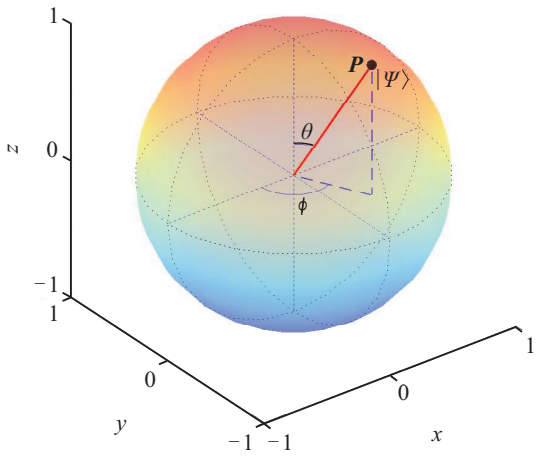

图 2　量子比特的 Bloch 球面描述

在量子计算中, 一个量子比特是一个可以在二维复希尔伯特空间中描述的两能级量子体系, 根据叠加原理, 量子比特的任何态都可以写成

$$|\varphi\rangle = \cos\frac{\theta}{2}|0\rangle + \mathrm{e}^{\mathrm{i}\phi}\sin\frac{\theta}{2}|1\rangle. \tag{8}$$

其中: $0 \leqslant \theta \leqslant \pi$, $0 \leqslant \phi \leqslant 2\pi$.

量子比特可以借助Bloch球面描述, 如图2所示. 由图2可知, Bloch球面上的任意一点 $P(x,y,z)$ 都与一个量子比特 $|\varphi\rangle$ 一一对应.

2.2 量子比特的绕轴旋转

在Bloch球面上, 量子比特的移动可通过绕着固定轴旋转实现, 旋转算子为一个酉矩阵. 量子比特绕着3个坐标轴旋转 δ 弧度的旋转矩阵分别为

$$\boldsymbol{R}_x(\delta) = \begin{bmatrix} \cos\dfrac{\delta}{2} & -\mathrm{i}\sin\dfrac{\delta}{2} \\ -\mathrm{i}\sin\dfrac{\delta}{2} & \cos\dfrac{\delta}{2} \end{bmatrix}, \tag{9}$$

$$\boldsymbol{R}_y(\delta) = \begin{bmatrix} \cos\dfrac{\delta}{2} & -\sin\dfrac{\delta}{2} \\ \sin\dfrac{\delta}{2} & \cos\dfrac{\delta}{2} \end{bmatrix}, \tag{10}$$

$$\boldsymbol{R}_z(\delta) = \begin{bmatrix} 1 & 0 \\ 0 & \mathrm{e}^{\mathrm{i}\delta} \end{bmatrix}. \tag{11}$$

令量子比特 $|\varphi\rangle = \left[\cos\dfrac{\theta}{2}, \mathrm{e}^{\mathrm{i}\phi}\sin\dfrac{\theta}{2}\right]^{\mathrm{T}}$, 则 $|\varphi\rangle$ 绕着3个坐标轴的旋转操作分别为 $|\varphi_x\rangle = \boldsymbol{R}_x(\delta)|\varphi\rangle$, $|\varphi_y\rangle = \boldsymbol{R}_y(\delta)|\varphi\rangle$, $|\varphi_z\rangle = \boldsymbol{R}_z(\delta)|\varphi\rangle$.

2.3 涡流中心初始化

不失一般性, 以最小值优化为例. 设优化空间为 D 维, 第 d 维取值范围为 $[\mathrm{Min}_d, \mathrm{Max}_d]$. 记 $\theta_d = \mathrm{rnd} \times \pi$, $\phi_d = \mathrm{rnd} \times 2\pi$, 则涡流中心可用量子比特初始化为

$$|\mu_0\rangle = [|\varphi_1\rangle, |\varphi_2\rangle, \cdots, |\varphi_D\rangle], \tag{12}$$

其中 $|\varphi_d\rangle = \left[\cos\dfrac{\theta_d}{2}, \mathrm{e}^{\mathrm{i}\phi_d}\sin\dfrac{\theta_d}{2}\right]^{\mathrm{T}}$, $d = 1, 2, \cdots, D$.

用泡利矩阵 σ_x、σ_y、σ_z 对 $|\mu_0\rangle$ 实施测量, 可以得到 $|\mu_0\rangle$ 中各量子比特在Bloch球面上的坐标.

$$x_d = \langle\varphi_d|\sigma_x|\varphi_d\rangle = \sin\theta_d\cos\phi_d, \tag{13}$$

$$y_d = \langle\varphi_d|\sigma_y|\varphi_d\rangle = \sin\theta_d\sin\phi_d, \tag{14}$$

$$z_d = \langle\varphi_d|\sigma_z|\varphi_d\rangle = \cos\theta_d. \tag{15}$$

其中: $\sigma_x = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$, $\sigma_y = \begin{bmatrix} 0 & -\mathrm{i} \\ \mathrm{i} & 0 \end{bmatrix}$, $\sigma_z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$.

由以上3式可知, $|\mu_0\rangle$ 对应Bloch球面上的3组坐标 $[x_1, x_2, \cdots, x_D]$、$[y_1, y_2, \cdots, y_D]$、$[z_1, z_2, \cdots, z_D]$, 每一组坐标对应一个优化解. 因此, 当迭代步数相同时, 本文方法的目标函数计算次数为普通涡流算法的3倍. 为使对比公平, 本文只采用 $[x_1, x_2, \cdots, x_D]$ 作为近似解. 因为 $x_d \in [-1, 1]$, 所以需要将其线性变换到 $[\mathrm{Min}_d, \mathrm{Max}_d]$, 该过程称为解空间变换, 变换式如下:

$$X_d = \frac{1}{2}[\mathrm{Min}_d(1 - x_d) + \mathrm{Max}_d(1 + x_d)]. \tag{16}$$

令 $\boldsymbol{X}_{\mathrm{best}} = [X_1, X_2, \cdots, X_D]$, 代入目标函数, 其目标值记为 F_{best}.

2.4 产生候选解

本文采用使 $|\mu_0\rangle$ 上的量子比特绕着坐标轴旋转的方法产生候选解. 关于坐标轴的确定, 采用 x 坐标作为候选解, 考虑到量子比特绕 X 轴旋转时, x 坐标不变, 因此本文选择 Y 轴或 Z 轴作为旋转轴. 令 $|\mu_0\rangle$ 上 $|\varphi_d\rangle$ 的Bloch坐标为 (x_d, y_d, z_d), 若 $|\varphi_d\rangle$ 绕 Y 轴旋转, 则旋转半径为 $\sqrt{x_d^2 + z_d^2}$; 若 $|\varphi_d\rangle$ 绕 Z 轴旋转, 则旋转半径为 $\sqrt{x_d^2 + y_d^2}$. 旋转半径越大对 x_d 的调整幅度越大, 因此, 如果 $|y_d| \leqslant |z_d|$, 则选择 Y 轴为旋转轴(如图3所示); 否则选择 Z 轴为旋转轴(如图4所示).

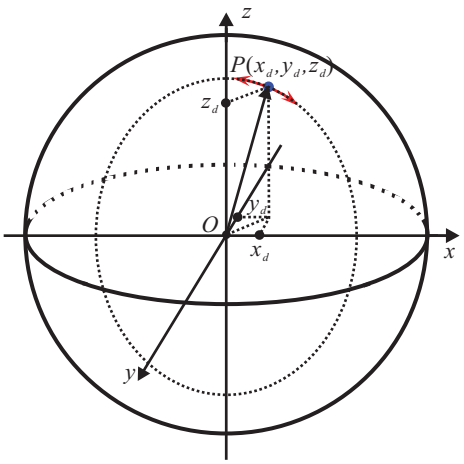

图3 量子比特绕 Y 轴旋转

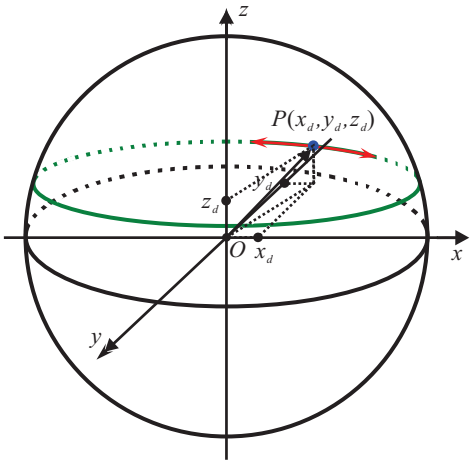

图4 量子比特绕 Z 轴旋转

关于 $|\varphi_d\rangle$ 的旋转角度, 采用中心为0、标准差为 $\sigma_d(t)$ 的高斯分布随机产生.

$$\delta_d(t) = \sigma_d(t) \times \mathrm{rand}\, n. \tag{17}$$

其中: $\mathrm{rand}\, n$ 为服从 $N(0,1)$ 分布的正态随机数, 标准差 $\sigma_d(t)$ 也采用文献[16]提出的逆不完全伽马函数方法逐代缩减, 具体表示为

$$\sigma_d(t) = \sigma_d(0)(1/\lambda)\gamma(\lambda, a_t). \tag{18}$$

$\sigma_d(0)$ 为标准差初值, $\lambda = 0.1$, $\gamma(\lambda, a_t)$ 按式(5)计算, a_t 按式(6)计算, t 为迭代步数.

关于$\sigma_d(0)$, 文献[16]中设置为搜索区间长度的一半. 因为本文采用量子比特在Bloch球面上绕坐标轴旋转实现搜索, $\sigma_d(0)$为旋转角度的标准差, 与$[\mathrm{Min}_d, \mathrm{Max}_d]$没有任何关系, 经过反复实验, 本文确定为$\sigma_d(0) = 0.1\,\mathrm{rad}$.

令候选解的个数为L, 根据$|y_d|$与$|z_d|$的大小关系确定旋转轴, 绕Y轴和Z轴的旋转操作分别为

$$|\widehat{\varphi}_d\rangle = \boldsymbol{R}_y(\delta_d(t))|\varphi_d\rangle, \tag{19}$$

$$|\widehat{\varphi}_d\rangle = \boldsymbol{R}_z(\delta_d(t))|\varphi_d\rangle. \tag{20}$$

将$|\mu_0\rangle = [|\varphi_1\rangle, |\varphi_2\rangle, \cdots, |\varphi_D\rangle]$上的$D$个量子比特都旋转后, 可生成1个候选解. 该过程重复L次, 即可得到L个候选解.

2.5 当前解更新

对L个候选解实施测量, 可得到这些解的Bloch坐标, 对每个候选解的x坐标实施解空间变换, 可得到L个实际解空间的优化解, 分别代入目标函数计算目标值.

令当代最优解为$\widehat{\boldsymbol{X}}_{\mathrm{best}} = [\widehat{X}_1, \widehat{X}_2, \cdots, \widehat{X}_D]$, 其目标值为$\widehat{F}_{\mathrm{best}}$, 对应的量子个体为$|\widehat{\mu}_0\rangle = [|\widehat{\varphi}_1\rangle, |\widehat{\varphi}_2\rangle, \cdots, |\widehat{\varphi}_D\rangle]$. 若$F_{\mathrm{best}} > \widehat{F}_{\mathrm{best}}$, 则置$|\mu_0\rangle = |\widehat{\mu}_0\rangle$, $\boldsymbol{X}_{\mathrm{best}} = \widehat{\boldsymbol{X}}_{\mathrm{best}}$, $F_{\mathrm{best}} = \widehat{F}_{\mathrm{best}}$.

以更新后的$|\mu_0\rangle$为中心, 以缩减后的$\sigma_d(t)$ $(d = 1, 2, \cdots, D)$为标准差随机产生旋转角度, 使$|\mu_0\rangle$绕着坐标轴旋转, 可产生下一代候选解. 循环此过程直到满足终止条件.

2.6 终止条件

终止条件可以设置为精度阈值, 即当最优解的目标值达到某种精度要求后迭代终止; 也可以设置为限定步数, 即当到达限定步数之后, 不论是否满足精度要求迭代均终止. 本文选择限定步数作为终止条件.

2.7 算法的收敛性

关于QIVS的收敛性, 给出如下结论.

定理1　当迭代步数足够多时, QIVS的迭代序列必定收敛.

证明　令每代候选解个数为L, 第n代涡流中心的量子比特描述为$|u_0(n)\rangle = [|\varphi_1\rangle, |\varphi_2\rangle, \cdots, |\varphi_D\rangle]$, 旋转角度的方差为$\sigma_n$, 则旋转角度可按下式产生:

$$\delta_i = \sigma_n \times \mathrm{rand}\, n_i,\ i = 1, 2, \cdots, L,$$

其中$\mathrm{rand}\, n_i$是均值为0、标准差为1的正态随机数. 根据QIVS候选解的产生原理, 当代第i个候选解$|u_i\rangle$可由下式产生:

$$|u_i\rangle = R(\delta_i)|u_0(n)\rangle,\ i = 1, 2, \cdots, L.$$

记上述L个候选解中最优解为$|u_j\rangle$, 根据QIVS的候选解更新原理, 若$|u_j\rangle$优于$|u_0(n)\rangle$, 则$|u_0(n+1)\rangle = |u_j\rangle$, 否则$|u_0(n+1)\rangle = |u_0(n)\rangle$.

如果$|u_0(n+1)\rangle = |u_j\rangle$, 则$|u_0(n+1)\rangle = R(\delta_j)|u_0(n)\rangle = R(\sigma_n \times \mathrm{rand}\, n_j)|u_0(n)\rangle$. 根据QIVS的$\sigma_n$缩减原理, $\lim\limits_{n\to\infty} \sigma_n = 0$, 所以有$\lim\limits_{n\to\infty} |u_0(n+1)\rangle = R(0)|u_0(n)\rangle$. 由旋转矩阵定义式(10)和(11)可知, 转角为零的旋转矩阵等价于单位矩阵, 即$R(0) = I$, 所以$\lim\limits_{n\to\infty} |u_0(n+1)\rangle = |u_0(n)\rangle$. 因为旋转矩阵为酉矩阵, 所以所有量子比特不会脱离Bloch球面. 从而表明, 当$n \to \infty$时, 迭代序列$\{|u_0(1)\rangle, |u_0(2)\rangle, \cdots, |u_0(n)\rangle\}$必收敛于Bloch球面上一定点.

若$|u_0(n+1)\rangle = |u_0(n)\rangle$, 则相当于$|u_0(n+1)\rangle = R(0)|u_0(n)\rangle$, 同理可证迭代序列收敛. □

2.8 算法描述

本文算法沿用了文献[16]的思想, 即通过对选定的涡流中心施加服从高斯分布且方差逐代减小的随机增量产生候选解集, 然后在候选解集中确定新的涡流中心, 直到满足终止条件. 其中, 高斯分布函数方差的减小方式体现了探索与开发之间的合理平衡. 然而, 本文算法在实施方式上, 采用量子比特描述个体, 采用方差逐代减小的高斯分布随机产生旋转角度, 采用使量子比特绕着坐标轴旋转的方法建立候选解. 本文算法可简要描述如下.

Inputs: 涡流中心$|\mu_0\rangle$, 标准方差$\sigma_d(0)$, 候选解个数L, 分辨率常数λ, 限定步数MaxItr, 当前步数$t = 0$, 当前最优函数值$F_{\mathrm{best}} = \infty$.

Repeat: 用式(18)计算$\sigma_d^l(t)$, 用式(17)生成正态分布的随机转角$\delta_d^l(t)$, 将$|\varphi_d\rangle$绕Y轴或Z轴旋转$\delta_d^l(t)$得到$|\widehat{\varphi}_d^l\rangle$, 对$|\widehat{\varphi}_d^l\rangle$测量得到$\widehat{x}_d^l(t)$, $d = 1, 2, \cdots, D$, $l = 1, 2, \cdots, L$. 对$\widehat{x}_d^l(t)$实施解空间变换可得候选解$\widehat{X}_d^l(t)$. 记当代最优解为$|\mu_0(t)\rangle$, 对应函数值为$F_{\mathrm{best}}(t)$, 若$F_{\mathrm{best}} > F_{\mathrm{best}}(t)$, 则$|\mu_0\rangle = |\mu_0(t)\rangle, F_{\mathrm{best}} = F_{\mathrm{best}}(t)$. $t = t + 1$.

Until: $t \geqslant \mathrm{MaxItr}$.

Output: 迄今为止的最优解$|\mu_0\rangle$.

2.9 算法复杂度

假设优化空间为D维, 候选解个数为L, 迭代步数为MaxItr. 由算法描述可知, 在Inputs部分, 因为没有循环, 所以计算复杂度为$O(1)$. 在Repeat部分, 共有三重循环, 由里到外分别为维数、候选解个数、迭代步数. 根据式(6)和(18), 计算$\sigma_d^l(t)$需要3次乘法; 根据式(17), 计算$\delta_d^l(t)$需要1次乘法; 获得$|\widehat{\varphi}_d^l\rangle$需要1次矩阵乘法(等价于4次复数乘法); 根据式(13), 投影测量获得$|\widehat{x}_d^l\rangle$需要6次复数乘法; 根据式(16), 解空间变换获得$\widehat{X}_d^l(t)$需要3次乘法. 余下的最优解更新部分不涉及乘法, 可以忽略. 综上, 乘法计算次数为$17 \times \mathrm{MaxItr} \times L \times D$, 因为$\Theta(17 \times \mathrm{MaxItr} \times L \times D) = D$, 所以

该阶段的计算复杂度为$O(D)$. 综上, 整个QIVS的计算复杂度为$T(D)=O(\max(1,D))=O(D)$, 即QIVS的计算复杂度是问题维数的多项式函数.

3 对比实验

文献[16]采用4类共50个函数考察了VS的性能. 为了检验本文QIVS的性能, 在每类中选取4个作为测试函数, 并与普通涡流搜索VS、粒子群优化PSO、人工蜂群优化ABC对比. 4种算法均在64位操作系统, 主频2.40 GHz, 内存16.0 GB的工作站上运行.

3.1 测试函数

1) 单峰变量可分离函数.

$$f_1(\boldsymbol{X})=\sum_{i=1}^{30}(\lfloor x_i+0.5\rfloor)^2,\ x_i\in[-100,100];$$

$$f_2(\boldsymbol{X})=\sum_{i=1}^{30}(x_i)^2,\ x_i\in[-100,100];$$

$$f_3(\boldsymbol{X})=\sum_{i=1}^{30}(ix_i)^2,\ x_i\in[-10,10];$$

$$f_4(\boldsymbol{X})=\sum_{i=1}^{30}(ix_i)^4+\text{random}[0,1),\ x_i\in[-1.28,1.28].$$

2) 单峰变量不可分离函数.

$$f_5(\boldsymbol{X})=\sum_{i=1}^{30}|x_i|+\prod_{i=1}^{30}|x_i|,\ x_i\in[-10,10];$$

$$f_6(\boldsymbol{X})=\sum_{i=1}^{30}\Big(\sum_{j=1}^{i}x_j\Big)^2,\ x_i\in[-10,10];$$

$$f_7(\boldsymbol{X})=\sum_{i=1}^{29}[100(x_{i+1}-x_i^2)^2+(x_i-1)^2],\quad x_i\in[-30,30];$$

$$f_8(\boldsymbol{X})=(x_1-1)^2+\sum_{i=2}^{30}i(2x_i^2-x_{i-1})^2,\quad x_i\in[-30,30].$$

3) 多峰变量可分离函数.

$$f_9(\boldsymbol{X})=\sum_{i=1}^{30}[x_i^2-10\cos(2\pi x_i)+10],\quad x_i\in[-5.12,5.12];$$

$$f_{10}(\boldsymbol{X})=\sum_{i=1}^{30}-x_i\sin(\sqrt{|x_i|}),\ x_i\in[-500,500];$$

$$f_{11}(\boldsymbol{X})=-\sum_{i=1}^{5}\sin(x_i)(\sin(ix_i^2/\pi))^{20},\ x_i\in[0,\pi];$$

$$f_{12}(\boldsymbol{X})=-\sum_{i=1}^{10}\sin(x_i)(\sin(ix_i^2/\pi))^{20},\ x_i\in[0,\pi].$$

4) 多峰变量不可分离函数.

$$f_{13}(\boldsymbol{X})=\frac{1}{4000}\sum_{i=1}^{30}x_i^2-\prod_{i=1}^{30}\cos\Big(\frac{x_i}{\sqrt{i}}\Big)+1,\quad x_i\in[-600,600].$$

$$f_{14}(\boldsymbol{X})=-20\exp\Big(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{30}x_i^2}\Big)-\exp\Big(\frac{1}{n}\sum_{i=1}^{30}\cos(2\pi x_i)\Big)+20+\mathrm{e},\quad x_i\in[-32,32].$$

$$f_{15}(\boldsymbol{X})=\frac{\pi}{n}\Big\{10\sin^2(\pi y_1)+\sum_{i=1}^{29}(y_i-1)^2\times[1+10\sin^2(\pi y_{i+1})]+(y_n-1)^2\Big\}+\sum_{i=1}^{30}u(x_i,10,100,4);$$

$$u(x_i,a,k,m)=\begin{cases}k(x_i-a)^m,\ x_i>a;\\0,\ -a\leqslant x_i\leqslant a;\\k(-x_i-a)^m,\ x_i<-a;\end{cases}$$

$$x_i\in[-50,50],\ y_i=\frac{1}{4}(x_i+1).$$

$$f_{16}(\boldsymbol{X})=0.1\Big\{\sin^2(\pi x_1)+\sum_{i=1}^{29}(x_i-1)^2\times[1+\sin^2(3\pi x_{i+1})]+(x_n-1)^2\times[1+\sin^2(2\pi x_n)]\Big\}+\sum_{i=1}^{30}u(x_i,5,100,4),\quad x_i\in[-50,50].$$

单峰函数是指在指定的优化空间只有一个最优解的函数, 而多峰函数是指在指定的优化空间存在大量局部最优解的函数. 所谓变量可分离函数是指含有n个变量的函数能够表述为只含1个变量的n个函数之和, 而变量不可分离函数不能表述为单变量函数之和的形式. 显然, 单峰变量可分离函数最容易优化, 而多峰变量不可分离函数是最难优化的.

3.2 参数设置

与VS相比, 尽管QIVS采用了完全不同的编码和候选解产生机制, 但控制参数并未增加. QIVS与VS一样, 也只有两个控制参数, 即候选解数和限定迭代步数. 为使对比公平, 两种算法采用相同的候选解数$L=50$; PSO和ABC的种群规模也取为50. PSO的控制参数取$c_1=c_2=2.0$, 惯性因子从0.9到0.4随迭代步数线性减小. ABC的控制参数$\text{limit}=100$. 为降低优化结果的随机性, 对于每个函数, 分别用4种算法独立优化30次, 以30次优化的平均结果作为对比指标.

3.3 结果对比

以30次独立优化结果的均值、单步迭代的平均时间作为对比指标, 4种算法在限定步数为100时的优化结果对比如表1所示.

表 1　限定步数为 100 时 30 次优化的平均结果对比

序号	QIVS		VS		PSO		ABC	
	均值	时间/10^{-3}	均值	时间/10^{-3}	均值	时间/10^{-3}	均值	时间/10^{-3}
f_1	15.600	1.666 0	14.000	0.228 8	2 218.0	0.421 0	1 466.0	0.567 8
f_2	2.295 5	1.661 4	2.729 6	0.239 2	2 201.0	0.443 9	1 160.0	0.537 4
f_3	5.338 5	1.654 9	8.467 7	0.244 4	242.51	0.425 6	177.54	0.554 3
f_4	0.072 9	1.689 4	0.130 6	0.639 6	1.098 0	0.727 6	1.093 5	0.713 2
f_5	1.711 3	1.671 2	3.502 0	0.150 8	20.036	0.416 4	2.606 4	0.730 1
f_6	433.39	1.669 2	3 523.0	0.473 2	22 246	0.604 0	36 719	3.021 7
f_7	124.92	1.6569	2 100.0	0.296 4	6.94e+5	0.453 0	63 380	0.689 5
f_8	7.343 6	1.659 5	14.270	0.223 6	5 726.0	0.407 3	930.17	0.655 7
f_9	22.641	1.667 9	98.018	0.312 0	249.83	0.471 3	85.709	0.682 8
f_{10}	−3 018	1.662 1	−9 637	0.327 6	−8 621	0.544 5	−868 1	0.753 7
f_{11}	−4.191	0.311 4	−4.251	0.254 8	−4.565	0.338 6	-4.685	0.452 9
f_{12}	−8.101	0.585 0	−8.756	0.343 2	−8.022	0.416 4	−9.142	0.540 8
f_{13}	0.880 2	1.665 3	0.971 9	0.306 8	20.689	0.507 9	21.741	0.828 1
f_{14}	2.145 1	1.665 3	2.161 8	0.291 2	10.800	0.503 4	13.428	0.848 4
f_{15}	4.502 7	1.671 2	65.608	0.696 8	69.962	0.846 6	8.192 1	1.703 5
f_{16}	0.632 6	1.670 5	7.354 5	0.592 8	136.54	0.787 1	7.723 6	0.936 3

由表 1 可知: 就优化结果而言, 对于 4 个单峰可分离函数, QIVS 优于 VS 的有 3 个, 优于 PSO 和 ABC 的均为 4 个; 对于 4 个单峰不可分离函数, QIVS 均优于 VS、PSO、ABC; 对于 4 个多峰不可分离函数, QIVS 也均优于 VS、PSO、ABC; 对于 4 个多峰可分离函数 ($f_9 \sim f_{12}$), QIVS 的优化性能不够理想, 优于 VS 和 ABC 的只有 1 个, 优于 PSO 的只有 2 个. 就运行时间而言, 首先将 16 个函数的单步迭代平均时间取平均值, 然后将该平均值作为对比指标. QIVS 的平均时间为 VS 的 5.060 6 倍, 为 PSO 的 3.001 1 倍, 为 ABC 的 2.019 4 倍. QIVS 较长的运行时间源于其涉及了量子比特的旋转、测量、解空间变换操作.

表 1 的结果仅表明, 在相同的迭代步数下, QIVS 展现了优于其他算法的性能. 为了增强对比结果的客观性, 下面给出 4 种算法基于相同运行时间的优化结果对比. 具体方案为, 首先将 4 种算法的迭代步数均取为 500, 然后再将 VS、PSO、ABC 的迭代步数分别设置为 2 500、1 500、1 000. 对比结果如表 2 所示.

表 2　相同运行时间下 30 次优化的平均结果对比

序号	QIVS	VS		PSO		ABC	
	500	500	2 500	500	1 500	500	1 000
f_1	3.400 0	5.000 0	5.000 0	4.800 0	0.200 0	13.600 0	0.200 0
f_2	0.000 0	0.000 0	0.000 0	0.885 3	0.000 1	2.057 8	0.000 0
f_3	0.000 0	0.021 5	0.000 1	0.156 2	0.000 0	0.076 5	0.000 0
f_4	0.010 4	0.061 0	0.026 0	0.084 1	0.052 7	0.614 4	0.302 4
f_5	0.000 1	0.011 1	0.000 1	0.276 0	0.000 5	0.234 1	0.005 4
f_6	7.8e-4	13.252	0.070 1	7.3e+3	4.1e+3	2.7e+4	1.7e+4
f_7	52.737	117.89	56.445	514.60	101.16	301.84	37.789
f_8	0.667 2	1.139 8	0.770 2	8.649 2	2.241 5	7.985 7	1.934 7
f_9	30.246	88.352	70.044	74.494	57.059	32.377	5.129 5
f_{10}	−4.7e+3	−9.3e+3	−1.0e+4	−1.0e+4	−1.1e+4	−1.0e+4	−1.1e+4
f_{11}	−4.279	−4.403	−4.586	−4.646	−4.671	−4.687	−4.687
f_{12}	−8.719	−8.326	−7.875	−8.924	−9.131	−9.511	−9.655
f_{13}	0.007 9	0.004 9	0.010 3	0.663 3	0.012 1	0.691 7	0.020 3
f_{14}	0.000 0	1.166 9	0.000 0	1.638 2	0.007 1	5.911 8	0.055 1
f_{15}	0.125 5	10.756	10.717	3.229 0	0.100 0	0.299 4	0.000 0
f_{16}	0.000 0	0.000 0	0.000 0	0.279 7	0.000 0	0.031 6	0.000 1

由表 2 可知: 在相同迭代步数时, 对于单峰变量可分离和不可分离函数 ($f_1 \sim f_8$), QIVS 均优于 VS、PSO、ABC; 对于多峰变量不可分离函数 ($f_{13} \sim f_{16}$), QIVS 均优于 PSO 和 ABC, 只有 f_{13} 劣于 VS; 对于多峰变量可分离函数, QIVS 与 VS 差别不大, 但劣于 PSO 和 ABC. 这些与表 1 的结论是一致的.

在相同运行时间时, QIVS 也表现出了相同的趋势. 即对于单峰 (变量可分离和不可分离) 和多峰变量不可分离函数, QIVS 明显优于 VS、PSO、ABC; 而对于多峰变量可分离函数, QIVS 的性能不够理想, 优于 VS、PSO、ABC 的函数分别为 2 个、1 个、0 个.

对于实验结果呈现出的 QIVS 优势, 给出如下的理论分析. QIVS 具有较强寻优能力的根本原因在于它的编码机制和候选解产生机制. 第一, 对于编码机制, QIVS 采用基于 Bloch 球面描述的量子比特编码, 采用量子比特的 x 坐标作为优化解. 对于第 i 维最优

解 x_i, 在 Bloch 球面上, 存在一个圆心在 $(x_i, 0, 0)$, 且与 X 轴垂直的半径为 $\sqrt{1-x_i^2}$ 的圆周 C, 该圆周上所有点的量子比特, 其 x 坐标都等于 x_i, 如图 5 所示. 这样一来, 就把 X 轴上的一个最优解映射到了圆周 C 上的所有量子比特, 这相当于极大地拓展了最优解的个数, 只要搜索到圆周 C 上的任意一点, 就等价于找到了该维的全局最优解, 因此可以增加逼近最优解的速度和获得全局最优解的概率. 第二, 对于候选解产生机制, QIVS 也采用基于高斯分布的随机数产生候选解, 并采用逆不完全伽马函数缩减搜索半径, 因此具有 VS 的优点. 然而 QIVS 仅是用高斯分布的随机数生成旋转角度, 具体采用使充当涡流中心的量子比特绕着坐标轴旋转的方法产生候选解. 由于采用了根据量子比特的两个坐标值 y_i 和 z_i 自适应选择旋转轴的方法, 可以保证在相同的转角下, 使对 x_i 的调整尽量大, 这有助于增强候选解的多样性, 从而在一定程度上能够克服 VS 在算法后期易于陷入早熟收敛的缺陷. 综合以上两方面, 量子比特编码和量子比特的绕轴旋转, 有效提高了 QIVS 算法的搜索能力.

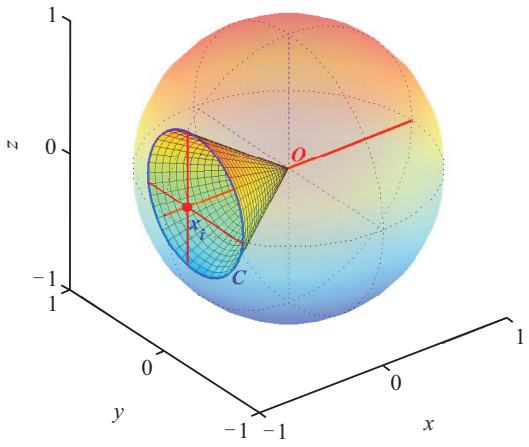

图 5 Bloch 球面上与 x_i 对应的圆周

4 结 论

本文提出了一种改进的涡流搜索算法, 该算法是最新提出的涡流算法的量子衍生版本. 算法采用量子比特对涡流中心编码, 采用逆不完全伽马函数调整高斯分布的方差, 采用高斯分布的随机数产生旋转角度, 采用量子比特的绕轴旋转产生候选解. 4 类标准函数极值优化的实验结果表明, 所提出的算法对于单峰优化和多峰变量不可分离的优化问题, 具有良好的适应性, 同时也揭示出量子比特编码和绕轴旋转, 能够切实提高目前涡流算法的搜索能力.

参考文献(References)

[1] Talbi E G. Metaheuristics: From design to implementation[M]. New Jersey: Wiley & Sons, 2009: 323-325.

[2] Boussaid I, Lepagnot J, Siarry P. A survey on optimization metaheuristics[J]. Information Science, 2013, 237(7): 82-117.

[3] Kirkpatrick S, Gelatt C D, Vecchi M P. Optimization by simulated annealing[J]. Science, 1983, 220(5): 671-680.

[4] Cerny C. Thermodynamical approach to the traveling salesman problem: an efficient simulation algorithm[J]. J of Optimization Theory and Application, 1985, 45(1): 41-51.

[5] Glover F. Future paths for integer programming and links to artificial intelligence[J]. Computers and Operations Research, 1986, 13(5): 533-549.

[6] Lourenco H R, Martin O, Stutzle T. Iterated local search: Framework and applications[J]. Int Series in Operations Research & Management Science, 2010, 146(5): 363-397.

[7] Alsheddy A. Empowerment scheduling: A multi-objective optimization approach using guided local search[D]. England: School of Computer Science and Electronic Engineering, University of Essex. 2011: 1-89.

[8] Hansen P, Mladenovic N, Perez J A M. Variable neighbourhood search: Methods and applications[J]. Annals of Operations Research, 2010, 175(1): 367-407.

[9] Storn R, Price K. Differential evolution - A simple and efficient adaptive scheme for global optimization over continuous spaces[R]. Berkley: Int Computer Science Institute, 1995: 1-15.

[10] Storn R, Price K. Differential evolution - A simple and efficient heuristic for global optimization over continuous spaces[J]. J of Global Optimization, 1997, 11(4): 341-359.

[11] Holland J H. Adaptation in Natural and Artificial Systems[M]. Ann Arbor: University of Michigan Press, 1975: 1-211.

[12] Dorigo M. Optimization, Learning and Natural Algorithms[D]. Italie: Dipartimento di Elettronica, Politecnico di Milano, 1992: 1-93.

[13] Kennedy J, Eberhart R C. Particle swarm optimization[C]. Proc of IEEE Int Conf on Neural Networks. New York: IEEE Press, 1995: 1942-1948.

[14] Dervis K, Bahriye B. A powerful and efficient algorithm for numerical function optimization: Artificial bee colony(ABC) algorithm[J]. J of Global Optimization, 2007, 39(3): 459-471.

[15] Dervis K, Bahriye B. On the performance of artificial bee colony(abc) algorithm[J]. Applied. Soft Computing, 2008, 8(1): 687-697.

[16] Berat D, Tamer O. A new metaheuristic for numerical function optimization: Vortex search algorithm[J]. Information Sciences, 2015, 293(1): 125-145.

(责任编辑：齐 霁)