跳跃逃逸算法优化模糊控制

胡劲松
(华南理工大学计算机学院 广州 510640)

摘 要 提出一种智能启发式搜索算法:跳跃逃逸算法,该算法使搜索直接从局部极小中跳出,而不是像模拟退火、禁忌搜索等方法从局部极小中慢慢地爬出,因此该方法能更快、更有效地解决局部极小问题。模糊控制器的优化是一个复杂的问题,目前的优化方法通常需要人的经验或只针对某些特殊的给定模型。结合跳跃逃逸算法和局部连续模糊算法构成了自优化模糊计算机控制系统,其优化过程自动完成,无需人的经验且不依赖任何模型。仿真实验表明该系统效果良好。

关键词 优化算法,局部极小,跳跃逃逸

中图法分类号 TP301.6 **文献标识码** A

Jumpily-escaping Algorithm Optimizes Fuzzy Control

HU Jing-song
(School of Computer Science and Engineering, South China University of Technology, Guangzhou 510641, China)

Abstract An intelligent heuristic search, called jumpily-escaping algorithm, was presented. The algorithm lets the current search "jum Pout", from the current local minimum by exploiting a new area that is far away from local minima obtained erenow, but not as Simulating Annealing or Tabu Search lets the current search "climb out", step by step, hardly and wanderingly. Therefore our method to solve local minimum problem is more successful and faster than other methods. The optimization of fuzzy controller is a complicated question. Most present optimization methods need experience or just fit for some special models. The self-optimum fuzzy controller based on the jumpily-escaping algorithm and the partly continuous fuzzy algorithm realizes automatic optimization without any experience or model. Simulated tests show that the system works well.

Keywords Optimizing algorithm, Partly extreme, Jumpily-escaping

许多智能优化算法,如遗传算法(GA)、进化规划(EP)、进化策略(ES)、模拟退火(SA)、禁忌搜索(TS)、微粒群(PSO)等日益受到重视。这些方法的应用十分广泛,取得了丰硕的成果。但是这些算法有一个致命的缺点:运算时间太长,无法满足实时优化、大规模优化的需要。如自动控制、网络优化等要求实时快速优化,这些智能算法的优化速度远不能满足要求。这些算法为了克服局部极小问题,运用了不同的策略,如模拟退火的按退火概率接受机制、禁忌算法的Tabu表、遗传算法的大规模种群,一方面这些方法有助于克服局部极小,另一方面也正是这些策略需要耗费大量的计算时间,造成运算时间长。为了适应实时、快速优化的需要,本文提出一种能快速克服局部极小的方法:跳跃逃逸算法。该算法使搜索直接从局部极小中跳出,而不是像模拟退火、禁忌搜索等方法从局部极小中慢慢地爬出,因此我们的方法能更快、更有效地解决局部极小问题。结合跳跃逃逸算法和局部连续模糊算法构成了自优化模糊控制器,其仿真效果优于模拟退火和遗传算法。跳跃逃逸算法也可广泛应用于其他需要快速优化的场合,如路由器算法、网络优化、飞行器和宇航控制等。

1 跳跃逃逸算法

对于智能优化算法而言,克服局部极小的能力是一个最重要的性能指标。尽管理论上有人证明遗传算法(GA)、模拟退火(SA)可以克服局部极小,但是其前提条件是搜索的时间无限长。在实际应用中,为了克服局部极小,这些方法不得不付出额外的巨大的计算量。

我们可以把优化搜索过程看成一个人或者多个人(搜索者)在一个有众多山峰山谷的区域,搜寻最低的山谷,克服局部极小的过程就是搜索者试图从一个山谷中爬出来,去寻找下一个山谷。对于模拟退火而言,当搜索者到达一个山谷,他并不是直接爬出,而是向上爬一步,向下退几步,其向下退的概率比向上爬要大。可以想象,如果是一个范围较大、底部较平的山谷,模拟退火从该山谷爬出来的概率是极小的。当然,理论上仍然存在爬出的可能。

禁忌算法是通过一张禁忌表,禁止搜索者走最近一段时期走过的旧路。当山谷的底部都走遍了,搜索者不得不向上开辟新的路径,从而慢慢地从山谷"盘旋式"地爬出来。其中

到稿日期:2008-06-04 本文受国家自然科学基金(60574078),广东省自然科学基金(31454),广州市科技计划应用基础项目(2006J1-C0321)资助。

胡劲松(1969—),男,博士,副教授,主要研究方向为智能优化算法、模糊控制、神经网络等,E-mail:cshjs@scut.edu.cn。

禁忌表的设计非常重要，如果表太小，不能禁止一个山谷的所有旧路，那么搜索者又会回到谷底。如果表太大，检索的速度非常慢且要占用大量的内存。显然，这两种算法都不是直接爬出山谷，实际上大量的计算量都是在爬出山谷的过程中消耗了。

实际上，优化速度和全局性往往是一对矛盾，经典的直接搜索法是直接"向下走/爬"，能比遗传算法、模拟退火等随机算法更快地发现局部极值点，而且精度更高，但是不能摆脱局部极小点，不具有全局优化能力。模拟退火、遗传算法有较好的全局性，但是其局部搜索速度慢，精度差。目前流行的方法是将遗传算法等与局部搜索法相结合，构成混合算法，一定程度上提高了精度，但是全局性并未改善。

本文首先用经典的直接搜索法（如模式搜索法）找到一个局部极小值点，然后继续找下一个极小值点。显然，要从一个新的起点开始，新的起点要尽量避开已找到的极值点及其邻域，否则找到的仍是旧的极值点。就像从一个漩涡中逃出来一样，已找到的极值点就是漩涡的中心，称之为跳跃逃逸算法。该算法与已有的各种克服局部极小值的机理完全不同，它通过巧妙地利用已找到的极值点为边界，形成新的起点，直接从局部极小值点跳出来。而禁忌搜索、模拟退火是先逐渐下坡找到局部极小点，然后逐渐爬坡，摆脱局部极小点。在爬坡过程中，不是连续向上爬，而是向上爬一点、向下退一点，反反复复，增加了许多额外的计算量。下坡的过程也是如此。由此可以看出，跳跃逃逸算法完全不存在局部极小问题，并且因为在下坡过程完成后，直接形成新的起点，而不是逐渐爬坡，摆脱上一个极小值，因此它没有爬坡过程，再加上其下坡时是连续向下的，计算量不止减少一半。模拟退火、Tabu 算法、遗传算法克服局部极小的机理增加了庞大的计算量，会使得算法的效率显著降低，并影响结果的精度。此外，跳跃逃逸算法尽量以未搜索过的区域为下一个搜索域，因此很少有遗漏和重复，而其他方法是随机选取，所以该算法的全局性强于遗传算法的随机初始群体，更强于模拟退火、禁忌搜索。

如上述，由于采用了直接搜索法，跳跃逃逸算法能更快地、精度更高地发现局部极点，同时通过"跳出"的方法避免了耗费大量的计算来克服局部极小，因此理论上本文的方法比其他方法更快，精度更高。具体方法如下：

假定我们已经发现了两个局部极小点 $X1$ 和 $X2$，那么下一个新的起点 $X\,start=(xs_1,xs_2,\cdots,xs_n)$ 就是在给定的搜索空间中离这两点的距离最远的点。假定我们发现了 k 个局部极小点：

$$X^{(1)}=(x_1^{(1)},\cdots,x_n^{(1)})$$

……

$$X^{(k)}=(x_1^{(k)},\cdots,x_n^{(k)})$$

令 $X\,last$ 是当前最后一个起始点：$X\,last=(xl_1,\cdots,xl_n)$，从该点我们找到了 $X^{(k)}$。

注意到我们的目标是构造一个新起始点，该点尽量远离目前已经发现的局部极小点，尤其是要远离最近发现的局部极小点。

计算从第 1 次局部极小点 $X_1^{(1)}$ 到第 $k-1$ 次的局部极小点 $X_1^{(k-1)}$ 的平均值：

$$Xav=(xav_1,\cdots,xav_n)$$

其中，

$$xav_1=\frac{1}{k-1}\sum_{i=1}^{k-1}x_1^{(i)}$$

$$xav_n=\frac{1}{k-1}\sum_{i=1}^{k-1}x_n^{(i)}$$

从 $X\,start$ 到 Xav，$X(k)$ 和 $X\,last$ 的距离分别是：

$$D1=\sqrt{(xs_1-xav_1)^2+\cdots+(xs_n-xav_n)^2}$$

$$D2=\sqrt{(xs_1-x_1^{(k)})^2+\cdots+(xs_n-x_n^{(k)})^2}$$

$$D3=\sqrt{(xs_1-xl_1)^2+\cdots+(xs_n-xl_n)^2}$$

令函数

$$f(X\,start)=f(xs_1,\cdots xs_n)=D1\cdot D2\cdot D3$$

当函数 $f(X_{\text{start}})$ 在给定的区域取得其极大值时，可以得到 $X\,start=(xs_1,xs_2,\cdots,xs_n)$。可以用经典的或近似的方法求 $f(X\,start)$ 的极大值，其表达式是已知的。注意：$f(X\,start)$ 不是要优化问题的目标函数，而是我们构造的为求一个较好起点的函数。

通过构造一个远离已发现极值点的新起点，跳跃逃逸算法使得搜索直接跳离了当前局部极小，因此该方法使得搜索永远不会停留在某个局部极小点，所以该方法在不付出额外搜索代价的前提下解决了局部极小这一难题。在搜索时间无限长的前提下，遗传算法、模拟退火、禁忌搜索可以克服局部极小，但是在实际应用中即使时间很长，也不容易得到全局最优解。

2 基于跳跃逃逸算法的自寻优模糊控制器

人的经验不可能是最优的，这也是模糊算法一直为人诟病的主要原因。因此模糊算法的优化一直是研究的热点，但是有较大的难度，迄今也没有被普遍采用。调整隶属函数和改变模糊规则是两类常用的方法，但是这两类方法本身就依赖人的经验，且实现起来复杂。采用智能优化算法，如遗传算法、模拟退火、微粒群等进行优化，理论上可以克服以上缺点，优化过程能基本自动化，但是这些策略需要耗费大量的计算时间，造成运算时间长。所以这些方法都不太适合实时的自寻优化控制，实际中难于应用。为了适应实时、快速优化的需要，本文采取了 2 个提高速度的措施：①采用优化速度更快的跳跃逃逸算法；②采用能快速响应同时又高精度的局部连续模糊算法。一起构成了自优化模糊控制器，实现快速优化和高精度控制。

2.1 局部连续模糊算法

查表型模糊算法和公式型模糊算法有静态误差，精度不高，但实时性好，占内存少。规则推理模糊算法精度高，但是推理复杂，不容易实现自动优化。局部连续模糊算法[1]是作者提出的一种简单快速的高精度方法，它通过对模糊查表法进行局部的控制面连续化，实现了高精度的模糊控制，同时避免了规则推理法的响应速度慢、对硬件要求高的缺点。

模糊控制器的输出 u 如下：

$$u=U(e^*,\Delta e^*)=\frac{R_1\times u_a+R_2\times u_b+R_3\times u_h+R_4\times u_d}{R_1+R_2+R_3+R_4} \tag{1}$$

因为静态误差产生的原因是在区间 $-0.5\leqslant e^*\leqslant 0.5$ 和 $-0.5\leqslant\Delta e^*\leqslant 0.5$ 控制量的输出 u 为 0，所以只需对 $|e^*|\leqslant 1$ 和 $|\Delta e^*|\leqslant 1$ 的区间进行连续化，因而大大节约了计算量，且精度很高，这正是局部连续法的关键所在[1]。

2.2 自优化系统的实现

模糊控制器的优化是比较困难的，目前主要有两种方式：优化模糊规则[2]和调整隶属函数[3]。这两种方法都很复杂，不容易实现，且优化不完全[4]。本文利用跳跃逃逸算法，同时对模糊表、量化因子 k_e 和 k_{ce}、比例系数 k_p、积分常数 k_i 进行同步优化。优化模糊表也就是优化模糊规则，调整量化因子也调整了隶属函数，所以本文的优化方法首先比前述两种方法全面。其次，因为采用了跳跃逃逸算法，其优化速度要快很多。尤其重要的是我们的方法不依赖人的经验，这种经验不是指模糊控制的经验，而是指优化模糊控制规则、隶属函数等与优化有关的经验。事实上，因为这种经验实际上是一种间接的作用，人们通常也不能提供好的直觉经验。此外，我们的方法具有普遍的适应性，不像一些改进的模糊优化算法，只是适用于某一类对象，或者需要对象模型的信息。由此可见，我们的方法更具有工程上的实用价值。具体的方法如下。

构造一个优化变量矩阵：

$$\boldsymbol{X}=\begin{bmatrix} & & & M & & & \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ k_p^* & k_i^* & k_e^* & k_{ce}^* & \times & \times & \times \end{bmatrix}$$

此处 M 是模糊表，其元素是区间[−4,4]内的整数。k_p^*，k_i^* k_e^* 和 k_{ce}^* 是区间[−30,30]内的整数，且存在以下关系式：

$$k_p=\lambda_1\times k_p^* \quad k_i=\lambda_2\times k_i^*$$

$$k_e=\lambda_3\times k_h^* \quad k_{ce}=\lambda_4\times k_{ce}^*$$

此处 $\lambda_1-\lambda_4$ 是实常数，其作用是将实际系统的取值范围变换到优化算法容易处理的范围。

例如，实际系统中 k_p 的范围是 $-5\leqslant k_p\leqslant 5$，而 k_p^* 是[−10,10]范围内的整数，那么 λ_1 可以由下式算出：

$$k_p=\lambda_1\times k_p^* \Rightarrow \lambda_1=k_p/k_p^*=0.5$$

目标函数：$J=\int_0^{+\infty} t\cdot|e|\cdot \mathrm{d}t$

e 和 X 之间的函数关系是复杂且未知的，因此 J 和 X 之间的函数关系也是未知的且不可分离的。但是 e 可以由被控对象的输出测量得到，目标函数可以间接计算。这些条件限制了传统解析法的使用，却正是智能优化算法所擅长的。

将目标函数离散化，并加上一个惩罚量，以限制超调量，于是最后的目标函数变为：

$$J=\sum_{i=1}^{N} i\cdot T_s^2|e(i)|+J_p$$

此处 T_s 和 i 分别代表采样时间和第 i 次采样。

图 1 是自优化模糊计算机控制系统，模块 1 是基于局部连续法的模糊控制器。μ_1 模块对 e^*，Δe^* 进行模糊化，其输出为 v，w 和连续因子 $R1$，$R2$，$R3$，$R4$。μ_2 是反模糊化模块。s 表示 $\mathrm{d}/\mathrm{d}t$。模块 2 是基于跳跃逃逸算法的自优化调节器。

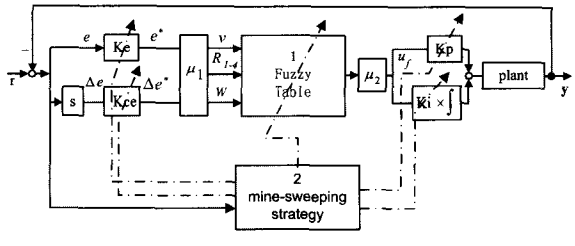

图 1 自优化模糊计算机控制系统(带箭头的虚线表示调节作用)

系统的主要流程如下：

(1) 模块 μ_1 对 e^*，Δe^* 进行模糊化，输出 v，w 和连续因子 $R1$，$R2$，$R3$，$R4$。由模块 1 和 μ_2 给出控制量 u_f。

(2) 如果系统的输出 y 在给定的时间达到一个稳定值，步骤转(3)，否则转(1)。

(3) 如果代价函数不满足要求，模块 2 对 k_{ce}，k_e，k_p，k_i 和模糊表进行调节，之后转步骤(1)，否则转步骤(4)。

(4) 优化过程结束，模块 2 的工作完成。系统按优化后的参数运行。

3 仿真和比较

给定如下非线性对象[5,6]：

$$\ddot{y}-\sin\dot{y}+(0.1\sin\frac{\pi}{2}t+1.2\dot{y})\cdot y^2-(1+0.3y)\cdot u=0$$

其中 y 是对象的输出，u 是对象的输入。$\dot{y}$ 和 $\ddot{y}$ 是 y 的一阶和二阶偏导。期望输出为标准阶跃响应。这是一个非常难于控制的对象。首先它是强非线性的，其次它的参数是随时间而变化的，尤为困难的是我们假定对控制器而言，该对象的模型完全是未知的。传统的控制算法对此是无能为力的，模糊优化控制有较好的效果[5-7]。分别使用模拟退火、遗传算法和跳跃逃逸算法进行优化。表 1 列出了 10 次优化的平均结果。跳跃逃逸算法的目标值 J 最小，效果最好。遗传算法的结果和跳跃逃逸算法差不多，但是其所用的评估次数(即优化时间)为跳跃法的 240 倍。模拟退火时间少于遗传算法，约为跳跃法的 2 倍，但是其结果不佳。跳跃逃逸算法实现了快速性和全局性的统一。

表 1 三种方法优化模糊控制器的平均结果和代价

	跳跃逃逸算法	模拟退火	遗传算法
J 的平均结果	112.3830	372.0022	113.4839
平均函数评价次数	41619	79179	10,000,500

图 2 中的虚线是非线性对象本身的阶跃响应，是发散且振荡的，显然该对象是难以控制的。图 2 中的实线是经跳跃逃逸算法优化后的模糊控制系统的输出，该曲线上升快，而几乎无超调，这是传统 PID 法无法实现的。在 3.5s 时加入一个干扰，系统波动很小。当然，因为该对象的参数是随时间做正弦振荡，非常难于控制，所以曲线有很小范围的缓慢波动。

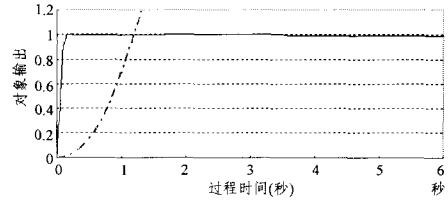

图 2 非线性对象的响应

结束语 由于采用了直接搜索法，跳跃逃逸算法能更快地、精度更高地发现局部极点，同时通过“跳出”的方法避免了耗费大量的计算来克服局部极小，因此理论上本文的方法比其他方法更快，精度更高。结合跳跃逃逸算法和局部联系模糊控制算法，构成了自优化模糊控制器，实现了对一个非线性参数时变对象的控制，效果良好。目前对于模型未知的非线性对象，还没有成熟的方案，本文对此做了有益的探索。

参 考 文 献

[1] Hu Jing-song, Zheng Qilun, Penghong, et al. Realizing fine

quality real-time fuzzy control by a simple algorithm[J]. Cybernetics and Systems, 2000, 31(7): 787-802

[2] Woodard S E, Garg D P. A numerical optimization approach for tuning fuzzy logic controllers[J]. IEEE Trans. Syst., Man, and Cybern, 1999, 29(4): 565-569

[3] Tao C W, Taur J. Design of fuzzy controllers with adaptive rule insertion[J]. IEEE Trans. Syst., Man, and Cybern., 1999, 29(3): 389-397

[4] Ying Hao. Some issues in fuzzy control theory and application [J]. Acta Automatic Sinica, 2001, 27(4): 590-592

[5] Hu Jing-song, Zheng Qilun. Time-varying modifying factor two layers self-optimizing fuzzy controller[J]. Acta Automatic Sinica, 2002, 28(6): 1006-1011

[6] Hu Jing - song, Zheng Qilun. Self - optimizing fuzzy controller based on extreme evolution algorithm[J]// Proceeding of IEEE 2003 Congress on Evolutionary Computation. Canberra, Australia, 2003: 2673-2677

[7] Hu Jing-song, Zheng Qilun. Time-varying modifying factor partly continuous fuzzy controller// Proceedings[J]. The IEEE International Conference on Fuzzy Systems FUZZ-IEEE 2003. St. Louis, MO, USA, 2003: 364-367

(上接第 170 页)

STEP 2　用 SUSAN 方法计算出数组 BP 中的角点。

STEP 3　通过 2.2 节中的方法确定数组 BP 中相邻两个角点之间的普通控制点。

STEP 4　依次取出数组 BP 中的控制点，其中角点取值 3 次，普通控制点取值 1 次；最后在控制点序列的首末各附加控制点 1 个。

STEP 5　根据绘制需要对取出的控制点序列施加相应的平移旋转伸缩变换，然后利用 3 次 B 样条曲线对控制点序列进行拟合，得到矢量化图形。

5　实验结果

运用上述算法对各种类型的二值图像进行大量的试验，矢量化结果均与实际边界形状相吻合，有效实现了图像边界的矢量化。限于篇幅，仅选取一例说明，实验中各参数取值如下：$\Delta d=6$，$\Delta t=10$。图 4 是一幅图像的实验结果，其中图 4(a)是原图像，图 4(b)是边界跟踪的结果，图 4(c)是从边界上提取到的控制点，图 4(d)是按原图大小直接绘制的矢量化图，图 4(e)是原图像边界与矢量化图的匹配结果，图 4(f)是对控制点施加伸缩旋转变换后的矢量图，其中伸缩系数为 1.5 倍，顺时针旋转 60 度。从图 4(e)看，矢量化图与原图的边界几乎一致，矢量化效果好；从图 4(f)看，变换后得到的矢量化图仍然保持着很好的形状，没有出现大的形变，表明了算法有效可行。对于图 4(a)的外轮廓，原图像共有 1683 个边界像素，本文的算法总共提取 166 个控制点即可对其进行表示，其中角点 22 个，普通控制点 144 个，实现了利用少量信息表达图像信息的功能。

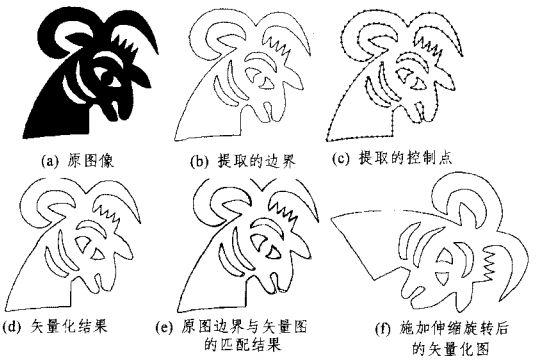

图 4　图像的矢量化

结束语　针对二值图像，本文提出了一种边界矢量化算法。算法的一个关键是提取控制点，控制点选取的恰当与否决定了矢量化效果的好坏，因此所提取的控制点应该能对边界的形状走向起到决定作用，本文所选取的角点和普通控制点达到了要求；其次，控制点的数量是影响算法性能的另外一个因素，通常控制点越多矢量化效果便越好，然而数据存储量和计算量也随之增大。关于阈值 Δd 和 Δt，取值越大所提取到的控制点越少，反之便越多；具体取值与实际图像的形状特点有关，形状变化平缓则 Δd 可取较小的值，曲线局部范围连续起伏少则 Δt 可取较大的值。下一步，将研究自适应的矢量化算法，实现根据边界的局部形状自动调整阈值大小，在保证矢量化效果的前提下使提取的控制点尽可能少。

参 考 文 献

[1] 沈立，张晨曦. 黑白图像的矢量化. 计算机辅助设计与图形学学报，2003，12(3)：170-173

[2] 王金鹤. 扫描图像曲线轮廓关键点的提取及其处理. 中国图像图形学报，2001，6(7)：699-702

[3] 余学军，彭立中. 二值图像曲线轮廓提取的新算法. 中国图像图形学报，2002，7(3)：272-275

[4] 沈萌红，崔云峰. 基于节点的曲线图表矢量化算法研究. 计算机工程与应用，2004(1)：96-98

[5] Ren Mingwu, Yang Jingyu, Sun Han. Tracing boundary contours in a binary image. Image and vision computing, 2002, 20(2): 125-131

[6] 张坤华，王敬儒，张启衡. 多特征复合的角点提取方法. 中国图像图形学报，2002，7(4)：319-324

[7] Freeman H, Davis L S. A corner finding algorithm for chain code curves. IEEE Trans. on Computers, 1997, 26: 297-303

[8] Wang H, Brady M. Real-time corner detection algorithm for motion estimation. Image and vision computing, 1995, 13(9): 695-703

[9] Smith S M, Brady J M. SUSAN-A new approach to low level image processing. Internal Technical Report TR95SMS1. Defence Research Agency, Chobham Lane, Chertsey, Surry, UK, 1995

[10] 孙家广，等. 计算机图形学(第三版)[M]. 北京：清华大学出版社，1998：308-318