文章编号: 1001-0920(2008)09-1068-05

基于粗细粒交叉的搜索算法

周 晖[1,2], 李丹美[1], 徐 晨[2], 邵世煌[1], 袁从明[2]
(1. 东华大学 信息科学与技术学院, 上海 201620; 2. 南通大学 电子信息学院, 江苏 南通 226019)

摘 要: 针对一种新的群集智能——自由搜索优化的不足,提出了基于粗细粒交叉的搜索算法. 该算法定义了粗粒交叉和细粒交叉两种算子. 通过粗粒交叉,有利于产生新的优秀个体,提高算法的全局搜索能力;采用细粒交叉,在搜索半径内产生更多的优良基因,提高局部搜索能力. 典型函数的实验结果表明:新算法的收敛速度、收敛精度、鲁棒性和稳定性大大优于基本自由搜索优化和标准微粒群算法.

关键词: 群集智能; 自由搜索优化; 微粒群算法; 交叉; 遗传算法
中图分类号: TP18 **文献标识码**: A

Free search algorithm based on coarse-grained and fine-grained crossover

ZHOU Hui[1,2], *LI Dan-mei*[1], *XU Chen*[2], *SHAO Shi-huang*[1], *YUAN Cong-ming*[2]
(1. College of Information Science and Technology, Donghua University, Shanghai 201620, China; 2. School of Electronics and Information, Nantong University, Nantong 226019, China. Correspondent: ZHOU Hui, E-mail: ntzhouhui@126.com)

Abstract: A novel free search algorithm based on coarse-grained and fine-grained cossover is proposed by combining free search with genetic algorithm. The algorithm defines two basis operators including coarse-grained cossover and fine-grained cossover. The operators of coarse-grained cossover make the algorithm obtain strong global exploring ability, and the operators of fine-grained cossover make the algorithm have strong local searching ability. Experimental results show that the convergence speed, the convergence probablity, the robustness and the stability of the algorithm are better than those of basic free search algorithm and particle swarm optimization.

Key words: Swarm intelligence; Free search(FS); Particle swarm optimization(PSO); Crossover; Genetic algorithm

1 引 言

群集智能是指“简单智能的主体通过合作表现出复杂智能行为的特性”,其概念源于对鸟群、蜜蜂、蚂蚁等群居生物群体行为的观察和研究,是简单生物群体的涌现现象的具体模式研究[1]. 蚁群算法(ACO)[2]和微粒群算法(PSO)[3]是群集智能的两种典型实现模式[4].

自由搜索(FS)优化是一种新的群集智能算法,它借鉴了自然界高等群居动物个体的生物特性:各异的感觉和运动能力[5]. 感觉和机动性分别被定义为灵敏度和搜索半径,并且利用动物信息素的机理,通过信息素和灵敏度的比较确定寻优目标[6]. 虽然这些研究还不够深入,但已显示出 FS 算法良好的寻优潜能.

FS 算法的个体采用有意识、自由、不确定的行为方式,这是与其他进化算法的最大区别. 它概念简单且容易实现,在寻优搜索的机理方面有较大的创新,并成功地应用于空中交通管理[7]和无线传感器网络节点定位[8].

研究发现,虽然 FS 优化比标准 PSO 算法的收敛速度和精度都有所提高,但该算法对搜索半径这一参数很敏感,设置不当将影响算法的性能,甚至陷入局部次优. 如何克服这一缺点,进一步挖掘 FS 的潜能,是一项有意义的工作.

本文在基本 FS 算法中融入遗传算法的交叉算子,提出了粗细粒交叉搜索算法(GAFS). 基于典型

收稿日期: 2007-05-31; **修回日期**: 2007-10-29.
基金项目: 国家自然科学基金项目(60474076); 江苏省高校自然科学基础研究项目(07KJB510095).
作者简介: 周晖(1963—),男,江苏南通人,副教授,博士生,从事智能优化、计算机网络等研究; 邵世煌(1938—),男,江苏苏州人,教授,博士生导师,从事智能系统与优化计算、生物信息技术等研究.

测试函数的仿真结果表明,新算法不仅解决了 FS 优化对搜索半径敏感的问题,而且在收敛速度、鲁棒性等方面,均比基本 FS 和标准 PSO 算法具有明显的优势.

2　粗细粒交叉的搜索算法

2.1　FS 算法简介

在搜索过程中,每个个体在其搜索半径 R_j 内随机行走 T 步,其进化方程如下:

$$\begin{cases} x_{tji} = x_{0ji} - \Delta x_{tji} + 2\Delta x_{tji} \cdot \mathrm{random}_{tji}(0,1), \\ \Delta x_{tji} = R_j(x_{i\max} - x_{i\min}) \cdot \mathrm{random}_{tji}(0,1). \end{cases} \tag{1}$$

其中: $R_j(0 < R_j \leq 1)$ 是搜索半径; $j(j = 1, 2, \cdots, m)$ 代表第 j 个个体; $i(i = 1, 2, \cdots, n)$ 为第 i 维变量; $\mathrm{random}_{tji}(0, 1)$ 是 $[0,1]$ 内均匀分布的随机数; x_{tji} 为搜索半径 R_j 内第 j 个个体第 t 步第 i 维分量; $t = 1,2, \cdots, T$; $x_{i\max}$ 和 $x_{i\min}$ 是第 i 维变量的最大值和最小值.

对目标函数的符号作如下规定:

$$f_{tj} = f(x_{tji}),\ f_j = \max(f_{ij}). \tag{2}$$

信息素定义为

$$P_j = f_j / \max(f_j), \tag{3}$$

其中 $\max(f_j)$ 是第 j 个个体的最佳值.

灵敏度定义为

$$\begin{cases} S_j = S_{\min} + \Delta S_j, \\ \Delta S_j = (S_{\max} - \Delta S_{\min}) \cdot \mathrm{random}_j(0,1). \end{cases} \tag{4}$$

其中 $S_{\min}$ 和 $S_{\max}$ 是灵敏度的最小值和最大值.

个体在 R_j 内按下式选择新的坐标点:

$$x_{0ji}' = \begin{cases} x_{0ji}', & P_k < S_j; \\ x_{ji}, & P_k \geq S_j. \end{cases} \tag{5}$$

即下一次搜索的起始点.

2.2　融入粗细粒交叉的搜索算法

分析式(1)可以发现,FS 算法中个体搜索行为的本质是在搜索半径 R_j 内的变异操作,而式(5)可视为该算法的选择算子. 变异的目的是:1)在搜索半径内调整个体的坐标值,使个体接近函数极值,属于局部搜索;2)维持种群的多样性,防止早熟. 若 R_j 较小,在局部作小步幅寻优,则"开发"能力强而"探索"能力弱;若 R_j 较大,在全局范围作大步幅探索,则反之. 可见, R_j 值的确定很重要,难度也很大,难以在全局搜索与局部寻优之间取得平衡. 另外,FS 算法的个体之间只能通过信息素进行间接通讯,而不能直接通讯.

本文在基本 FS 算法中嵌入了粗细粒交叉算子. 粗粒交叉在个体之间进行,增加个体之间直接通讯的渠道,同时利用子代个体与父代个体之间编码结构的相似性,保存父代的优良基因,因此粗粒交叉有利于生成包含更多优良基因的新个体,提高全局搜索能力. 细粒交叉则应用于搜索半径内部,个体产生更多的优良基因,经过选择提高局部寻优能力.

2.2.1　粗粒交叉算子

交叉是两个父代个体的部分结构加以替换重组而生成新个体的操作. 本文在实数编码的个体之间采用离散重组和多点交叉的方式. m 个交叉位置 K_i 可以无重复地随机选择,交叉点之间的变量间续地相互交换. 定义个体之间的这种交叉方式为粗粒交叉,具体操作如下:

设两个父代向量分别为

$$\begin{cases} X_j = (x_{j1}, x_{j2}, x_{j3}, x_{j4}, \cdots, x_{jl}, \cdots, x_{jn}), \\ X_k = (x_{k1}, x_{k2}, x_{k3}, x_{k4}, \cdots, x_{kl}, \cdots, x_{kn}). \end{cases} \tag{6}$$

执行粗粒交叉,产生两个新的后代

$$\begin{cases} X_j' = (x_{j1}, x_{k2}, x_{j3}, x_{k4}, \cdots, x_{kl}, \cdots, x_{jn}), \\ X_k' = (x_{k1}, x_{j2}, x_{k3}, x_{j4}, \cdots, x_{jl}, \cdots, x_{kn}). \end{cases} \tag{7}$$

上述交叉后,采用精英选择策略:若子代优于父代,则用子代代替父代;否则,仍保留父代.

粗粒交叉能保持交叉前后两个子代个体之间的欧几里德距离与两个父代个体之间的欧氏距离相等. 因为

$$\| X_j - X_k \| = \| X_j' - X_k' \| = \sqrt{(x_{j1} - x_{k1})^2 + (x_{j2} - x_{k2})^2 + \cdots + (x_{jn} - x_{kn})^2}, \tag{8}$$

所以相距较远的两个父代个体经过交叉产生的子代个体也相距较远,而相距较近的两个父代个体经过交叉产生的子代个体也相距较近. 这个规律既有利于保持种群的多样性,又能提高搜索效率. 开始优化时,个体之间差异较大;进行全局搜索时,随着寻优过程的展开,个体大多向最优区域趋近,因此个体之间差异逐渐变小. 式(8)说明,粗粒交叉不破坏基本 FS 算法的搜索格局.

交叉是遗传算法产生新个体的主要方法[9]. 粗粒交叉算子可充分实现个体之间的信息交换,提高全局搜索能力,同时减小计算开销.

2.2.2　细粒交叉算子

个体在搜索半径 R_j 内随机行走 T 步,如式(1)所示. 本文将这些坐标点视为基因,将适应值最优的个体直接保留复制到下一代,保证 FS 算法的基本操作产生的最优个体不会被交叉等操作破坏,其他个体之间采用多 α 算术重组交叉. 具体方法是随机选择两个作为父代,子代按下式产生:

$$\begin{cases} X_m' = \alpha_1 X_m + (1 - \alpha_1) X_i, \\ X_i' = \alpha_2 X_i + (1 - \alpha_2) X_m, \end{cases} \tag{9}$$

其中 α_1 和 α_2 是比例因子,由 $[0,1]$ 中均匀分布的随

机数产生.

相对于个体之间的粗粒交叉算子,本文定义上述交叉操作为细粒交叉算子. 细粒交叉既保留了精英个体,又能产生更多的优良基因,从而提高了算法的局部搜索能力. 从整体上看,新算法将 GA 与 FS 优化有机融合,交叉算子与变异算子相互配合,间接通讯和直接通讯同时作用,全局探索和局部搜索同时加强,不仅提高了算法的性能,而且使算法对参数设置的依赖性大大减弱,特别是对搜索半径敏感的问题得到了解决.

2.2.3 算法实现

GAFS 的算法流程是在基本 FS 算法的框架中增加了粗细粒交叉算子. 整个算法流程如下:

Step1: 初始化:

1) 设定种群规模 m,搜索代数 G,搜索步数 T和各个体的邻域半径 R_j;

2) 随机产生初始种群.

Step2: 寻优过程:

1) 粗粒交叉,具体操作见 2.2.1 节;

2) 个体 $j(j = 1,2,\cdots,m)$ 在搜索半径 R_j 内按式(1) 随机行走 T 步;

3) 细粒交叉,具体操作见 2.2.2 节;

4) 按式(3) 计算信息素 P_j;

5) 按式(4) 计算灵敏度 S_j;

6) 按式(5) 释放信息素 $P_j \rightarrow X_k$,选择搜索的下一轮起始点,$X_{0j}' = X_k(S_j, P_k)$.

Step3: 终止判断. 若满足最小阈值或搜索代数 g 达到设定代数,则执行 Step4;否则,转至 Step2.

Step4: 输出结果,结束运行.

3 算例与分析

为了验证 GAFS 优化算法的性能,选择一组 Benchmark 函数,分别用基本 FS,标准 PSO 和 GAFS 算法进行仿真实验,并对实验结果进行分析比较. 测试函数如下:

1) Sphere 函数

$$f_1(\vec{x}) = \sum_{i=1}^{n} x_i^2; \tag{10}$$

2) Rosenbrock 函数

$$f_2(\vec{x}) = \sum_{i=1}^{n} (100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2); \tag{11}$$

3) Rastrigin 函数

$$f_3(\vec{x}) = \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i) + 10); \tag{12}$$

4) Griewank 函数

$$f_4(\vec{x}) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{X_i}{\sqrt{i}}\right) + 1; \tag{13}$$

5) Schaffer's f_6 函数

$$f_6(\vec{x}) = 0.5 - \frac{(\sin\sqrt{x_1^2 + x_2^2})^2 - 0.5}{(1 + 0.001(x_1^2 + x_2^2))^2}. \tag{14}$$

测试函数的维数(Dim),取值范围(Range),最小误差阈值(Cri),全局最优解(Opt) 和最优点位置(Position) 如表 1 所示.

表 1 Benchmark 函数

F	Dim	Range	Opt	Cri	Position
f_1	30	$[-100,100]^n$	0	0.01	$(0,\cdots,0)$
f_2	30	$[-30,30]^n$	0	100	$(1,\cdots,1)$
f_3	30	$[-5.12,5.12]^n$	0	100	$(0,\cdots,0)$
f_4	30	$[-600,600]^n$	0	0.1	$(0,\cdots,0)$
f_6	2	$[-100,100]^2$	1	10^{-5}	$(0,\cdots,0)$

注: f_1 函数的 Cri: FS 和 PSO 取 0.01, GAFS 取 10^{-6}.

函数 f_1 是单峰二次函数; f_2 是很难极小化的病态函数; f_3 和 f_4 都是具有众多局部极小点的多峰函数; f_6 的全局最优点被次优点包围,一般算法很难找到全局最优点.

对上述函数从不同的初始解出发,每个函数分别采用 GAFS,FS,PSO 3 种算法,每种算法的种群规模分别取 $m = 30$ 和 $m = 60$,各自独立运行 20 次,比较其实验结果.

关于 FS 算法的参数设置,搜索半径 R_j 的大小反映了寻优范围,搜索步 T 的大小反映了在 R_j 半径内的搜索密度, R_j 的确定须兼顾全局和局部搜索, T 的大小须兼顾搜索精度和计算开销. 作者根据大量实验,总结出参数设置的一些规律:一般取 $T = m$; R_j 可分别取值为 1,0.5,0.1,其中 $R_j = 1$ 的个体数约为 $m/4 \sim m/3$; $R_j = 0.5$ 的个体数约为 $m/3 \sim m/2$; $R_j = 0.1$ 的个体数约为 $m/4 \sim m/3$. 在本次实验中, R_j 按下式确定:

$$R_j = \begin{cases} 1, & j = 1,\cdots,m/3; \\ 0.5, & j = m/3 + 1,\cdots,2m/3; \\ 0.1, & j = 2m/3 + 1,\cdots,m. \end{cases}$$

GAFS 算法的参数设置和 FS 算法的参数设置相同. PSO 的参数选取参照文献[10]:搜索代数 $G = 1000$,加速常数 $c_1 = c_2 = 1.7$,惯性权重 $w = 0.6$.

图 1 ~ 图 5 为种群数 $m = 30$ 时,分别采用 GAFS,FS,PSO 3种算法,对上述5种Benchmark函数在表 1 给定的维数、搜索范围、最小误差阈值条件下,运行 20 次的最佳适应度进化曲线. 限于篇幅, $m = 60$ 的最佳适应度进化曲线没有列出. 最佳适应度进化曲线是算法整个寻优搜索过程的客观反映. 图中的实验结果显示,无论是对单模态函数 f_1 和 f_2,

还是对多模态函数 f_3, f_4, f_6, GAFS 算法的收敛速度都明显快于其他两种算法. 图 1 说明,针对 Sphere 函数,算法在收敛精度要求大大高于其他两种算法时,其收敛速度仍然快于其他两种算法. 图 3 反映出, GAFS在对多模态函数的寻优过程中,能够通过

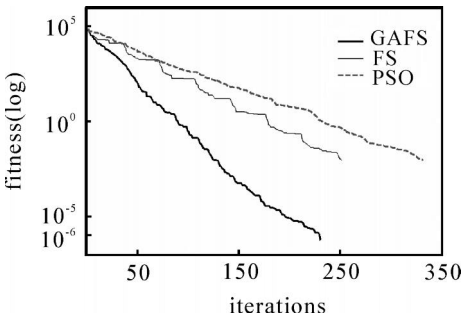

图 1　$f_1(x)$ 的最佳适应度进化曲线

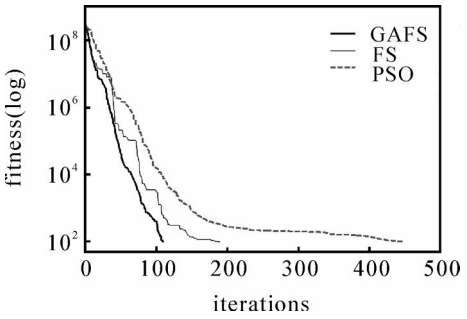

图 2　$f_2(x)$ 的最佳适应度进化曲线

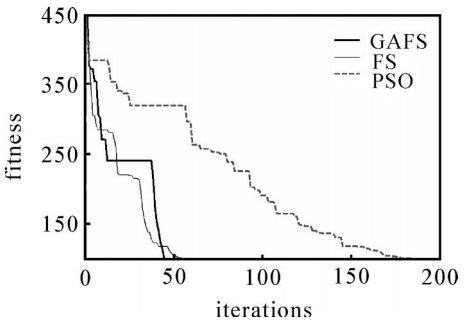

图 3　$f_3(x)$ 的最佳适应度进化曲线

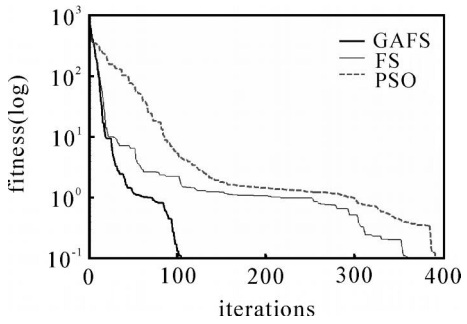

图 4　$f_4(x)$ 的最佳适应度进化曲线

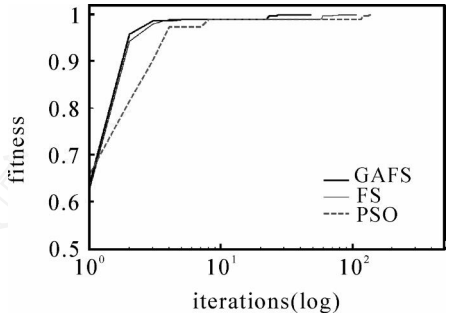

图 5　$f_6(x)$ 的最佳适应度进化曲线

粗粒交叉跳出局部次优,且以较快的速度实现收敛.

表 2 给出了种群数分别为 $m = 30$ 和 $m = 60$, 3 种算法对上述 5 种函数在 $G = 1000$ 时运行 20 次寻优,各函数的平均最优适应值和标准差. 表 2 的结果显示, GAFS 算法的平均最优适应值明显优于普通 FS 和标准 PSO 算法,说明其收敛精度比其他两种算法有较大提高. GAFS 算法的标准差有时比 FS 和 PSO 大,如对 f_2 和 f_3 的寻优结果,但从总的情况看, GAFS 的标准差不高于 FS 和 PSO. 另外,实验中还发现, GAFS算法对搜索半径的变化不敏感,只要 R_j 设置在 0.5 ～ 1 之间,对算法性能影响都不大.

由此可见,本文提出的 GAFS 算法整体寻优性能比基本FS算法有很大改善. 当然, GAFS算法每

表 2　3 种算法运行 20 次的函数平均最优解和标准差

F	m	Mean best fitness			Standard deviation		
		GAFS	FS	PSO	GAFS	FS	PSO
f_1	30	8.3679e-07	0.00948	0.009406	1.01640938e-07	5.21536e-04	3.5486e-04
	60	8.8536e-07	0.004448	0.0092374	0.92669466e-07	0.00367570	7.6620e-04
f_2	30	94.415	93.1995	98.73265	4.3605	6.150981698	1.3636
	60	93.7930	96.4768	98.54945	4.826688	3.70294466	2.0883
f_3	30	98.3980	98.7422	99.4613	2.4734	1.431287427	0.6221
	60	93.4011	97.796	98.71125	5.8974845	1.5767249	1.7451
f_4	30	0.0876	0.0929	0.0943515	0.0095	0.005134746	0.0081
	60	0.087334	0.0856428	0.09494245	0.00717523	0.012287487	0.0048
f_6	30	0.9999997	0.999998	0.99999436	2.6312643e-07	4.3035624e-06	2.85812e-06
	60	0.9999995	0.999996	0.99999441	2.6800039e-07	5.47722558e-06	2.86659e-06

一代搜索的运算量比基本 FS 略高,但它的收敛代数明显小于基本 FS,因此该算法的寻优效率仍高于 FS 算法.

4 结 论

FS 算法是一种新的群集智能优化算法,本文针对其存在的不足,提出了粗细粒交叉搜索算法,将遗传算法中的交叉算子融合到 FS 算法. 通过个体之间的粗粒交叉和个体搜索半径 R_j 内部的细粒交叉,提高了算法全局搜索能力和收敛速度,并且实现了算法"探索"与"开发"的动态平衡. 通过对典型函数的测试验证了所提出算法的正确性和高效性.

参考文献(References)

[1] Colorni A, Dorigo M, Maniezzo V. Distributed optimization by ant colonies[C]. Proc 1st European Conf on Artificial Life. Pans: Elsevier, 1991: 134-142.

[2] Dorigo M, Blum C. Ant colony optimization theory: A survey[J]. Theoretical Computer Science, 2005, 344: 243-278.

[3] Kennedy J, Eberhart R. Particle swarm optimization [C]. IEEE Int Conf on Neural Networks. Piscataway: IEEE Service Center, 1995: 1942-1948.

[4] 黄芳, 樊晓平. 基于岛屿群体模型的并行粒子群优化算法[J]. 控制与决策, 2006, 21(2): 175-180.
(Huang F, Fan X P. Parallel particle awarm optimization algorithm with island population model[J]. Control and Decision, 2006, 21(2):175-180.)

[5] Penev K, Littlefair G. Free search — A comparative analysis[J]. Information Science, 2005, 172(1): 173-193.

[6] 周晖, 李丹美, 邵世煌, 等. 一种新的群集智能算法——自由搜索[J]. 东华大学学报, 2007, 33(5): 32-36.
(Zhou H, Li D M, Shao S H, et al. A novel swarm intelligent algorithm: Free search[J]. J of Donghua University, 2007, 33(5): 32-36.)

[7] Penev K. Adaptive computing in support of traffic management[J]. Adaptive Computing in Design and Manufacturing, 2004, 22(4): 20-22.

[8] Zhou Hui, Li Dan-mei, Shao Shi-huang, et al. A novel intelligent estimation algorithm in WSN location bsed on free search [C]. IEEE Int Conf on Wireless Communication of Networking and Mobile Computing. Shanghai, 2007: 2629-2632.

[9] 玄光男, 程润伟. 遗传算法与工程优化[M]. 北京: 清华大学出版社, 2005: 45-54.
(Mitsuo Gen, Cheng R W. Genetic algorithms and engineering optimization [M]. Beijing: Tsinghua University Press, 2005: 45-54.)

[10] Ioan Cristian Trelea. The particle swarm optimization algorithm: Convergence analysis and parameter selection[J]. Information Processing Letters, 2003, 85(6): 317-325.

(上接第 1067 页)

[4] Chiang T Y, Hung M, Yan J J, et al. Sliding mode control for uncertain unified chaotic systems with input nonlinearity[J]. Chaos Solitions & Fractals, 2007, 34(2): 437-442.

[5] Park J H. On synchronization of unified chaotic systems via nonlinear control[J]. Chaos, Solitons & Fractals, 2005, 25(3): 699-704.

[6] Lu J H, Chen G R, Cheng D, et al. Bridge the gap between the Lorenz system and Chen system[J]. Int J of Bifurcation and Chaos, 2002, 12(12): 2917-2926.

[7] Lu J A, Wu X Q, Lu J H. Synchronization of an unified chaotic system and the application in secure communication[J]. Physics Letter A, 2002, 305(16): 365-370.

[8] 王兴元, 段朝峰. 基于线性状态观测器的混沌同步及在保密通信中的应用[J]. 通信学报, 2005, 26(6):105-136.
(Wang X Y, Duan C F. Observer based chaos synchronization and its application to secure communication[J]. J on Communications, 2005, 26(6): 105-136.)

[9] 刘斌, 张曾科. 一种三对角结构非线性系统的稳定性及其应用[J]. 自动化学报, 2007, 33(4): 442-445.
(Liu B, Zhang Z K. Stability of nonlinear systems with tridiagonal structure and its applications[J]. Acta Automatica Sinica, 2007, 33(4): 442-445.)