DOI:10.19651/j.cnki.emt.2208820

改进差分进化算法优化多值属性系统诊断策略*

邱晓红 徐 聪
(江西理工大学软件工程学院 南昌 330000)

摘 要: 测试序列优化问题是故障诊断过程中的关键性问题;针对多值属性系统的测试序列优化问题,采用自适应差分进化算法,结合多值属性系统的特点,分析变异算子在算法中的作用,并设计了个体的编码策略以及两种不同的诊断方式,提出一种将高斯,柯西变异算子与多差分策略进行融合的差分进化算法;通过实验对比分析,结果表明该算法不仅可以很好的应用于多值属性系统,而且在处理二值属性系统的测试序列优化问题时,与已有算法相比,该算法得到的测试点数目更少,期望测试代价更低,可用于多值属性系统求解诊断策略问题。

关键词: 多值属性系统;测试序列优化;变异算子;差分进化算法
中图分类号: TP206.3 **文献标识码:** A **国家标准学科分类代码:** 520.3050

Improved differential evolution algorithm to optimize diagnosis strategy of multi-valued attribute system

Qiu Xiaohong Xu Cong
(School of Software Engineering, Jiangxi University of Science and Technology, Nanchang 330000, China)

Abstract: The test sequence optimization problem is a key problem in the fault diagnosis process; for the test sequence optimization problem of the multi-valued attribute system, the adaptive differential evolution algorithm is used, combined with the characteristics of the multi-valued attribute system, analysed the role of mutation operators in algorithms, and designed the individual coding strategy and two different diagnosis methods, proposed a differential evolution algorithm that integrates Gaussian, Cauchy mutation operators and multi-difference strategies; Through experimental comparison and analysis, the results show that the algorithm can not only be well applied to multi-valued attribute systems, but also when dealing with the test sequence optimization problem of binary attribute systems, compared with the existing algorithms, the number of test points obtained by this algorithm is less. It is expected that the test cost is lower, and it can be used to solve the problem of diagnosis strategy in multi-valued attribute systems.

Keywords: multi-valued attribute system; test sequence optimization; mutation operator; differential evolution algorithm

0 引 言

随着我国航天工程的不断发展,许多大型电子设备都需要依赖更为复杂的系统,测试性可提供故障状态的实时检测和故障的及时诊断,能够确保系统可靠且高效的运行。目前对于故障诊断策略[1-2]的研究也越来越广泛,测试序列优化问题(optimal test sequence problem, OTP)是根据测试与故障的关联性矩阵,设计一组最优测试序列,其满足指定故障检测率,并且总体的测试代价最小,该问题被证明是Np完全问题[3]。目前对于该问题的研究大多数都是针对二值属性系统的,对于故障状态的检测结果只有通过和不通过。在实际情况中,测试过程中测试往往是不完全可靠[4]的,很多系统的测试结果具有多个值,那么具有多个测试值的系统被称为"多值属性系统",其诊断策略的优化相对二值属性系统较为复杂,目前已有学者对于多值属性系统进行研究。

基于哈夫曼编码的思想,潘兴涛等[5]通过构建与或树启发式搜索算法来解决多值属性系统的OTP问题,其存储空间随系统规模成指数增长,容易出现计算爆炸的问题;黄以锋等[6]提出基于熵的启发式,根据当前测试点的最大熵值来选择测试点,其测试点的选择方式具有贪婪性,会引发陷入局部最优解从而找不到全局最优解的问题;基于此黄以锋等[7]进一步提出Rollout算法,Tian等[8]对Growing算法进行拓展得到近似最优的诊断策略。田恒等[9]通过改

收稿日期:2022-01-10
* 基金项目:江西省自然科学基金(20181BAB202018)、研究生创新专项资金项目(XY2021-S168)资助

进的蚁群算法对测试序列进行优化,得到较好的结果。

在处理优化问题时,差分进化算法相比其它群智能算法表现出较强的性能,例如个体具有速度属性的惯性速度差分进化算法(inertial velocity DE,IVDE)[10],通过随机组合试验向量生成策略和控制参数设置的复合差分进化算法(composite DE,CoDE)[11],具有 Pbest 引导机制的适应性多策略差分进化算法(AMSDE)[12],具有多个优秀 DE 变体的集合,从而推导出基于多种群框架的 EDEV[13]算法等;这些算法对目标函数的优化都取得了较好的成果,但目前差分进化算法还未应用到多值属性系统的 OTP 问题,基于此,本文提出将高斯,柯西变异算子融合多策略的差分进化(gauss-cauchy multiple strategies differential evolution,GCMDE)算法。针对多值属性系统的 OTP 问题,通过分析诊断策略问题模型、高斯,柯西变异算子的意义以及对 GCMDE 的算法的描述,选择不同算法进行实验对比,最后简要分析了两种测试序列诊断方式的优缺点,并给出一些具有可参考性的结论。

1 诊断策略问题描述

1.1 多值属性系统

多值 D 矩阵是多值属性系统中故障和测试之间的关系性矩阵,如表 1 所示。

表 1 多值 D 矩阵

故障状态	测试点 t_i		…		先验概率
	t_1	t_2	…	t_n	
	测试费用 c_i				
	c_1	c_1	…	c_n	
f_1	d_{11}	d_{12}	…	d_{1n}	p_{f_1}
f_2	d_{21}	d_{22}	…	d_{2n}	p_{f_2}
…	…	…	…	…	…
f_m	d_{m1}	d_{m2}	…	d_{mn}	p_{f_m}

其矩阵的行向量表示故障,$d_{ij}=y$ 表示为利用第 j 个测试检测第 i 个故障的检测结果,其结果值的大小表示检测该故障状态的严重程度;如果检测结果为 y,则表示第 i 个故障状态可能发生;y 的取值为正整数($0\leqslant y\leqslant k-1$),其中 k 表示为该多值属性系统取值的维度。

1.2 测试序列优化问题的描述

测试序列优化问题通常定义为五元组(F,P,T,C,D)。其具体描述如下:

1) $F=\{f_1,f_2,f_3,f_4,\cdots,f_m\}$ 为系统可能发生的 m 个故障状态集合,其中 f_1 在本文中表示为无故障状态,$f_i(1<i\leqslant m)$ 表示为第 i 个故障状态;

2) $P=\{p(f_1),p(f_2),p(f_3),\cdots,p(f_m)\}$ 为故障状态集 F 中各故障发生的先验概率,且集合 P 中的先验概率总和为 1;

3) $T=\{t_1,t_2,t_3,t_4,\cdots,t_n\}$ 为当前系统提供 n 个可用于检测故障发生的测试集;

4) $C=\{c_1,c_2,c_3,c_4,\cdots,c_n\}$ 为测试费用集,其中 c_i 表示使用测试 t_i 所花费的费用,包括所消耗的人力、物力以及时间等;

5) $D=[d_{ij}]_{m\times n}$ 表示为故障和测试之间的关联矩阵;

多值属性系统的最优测试序列问题就是依据所给的五元组,设计一组具有先后顺序的测试序列,通过使用测试序列中的测试点隔离出故障状态集 F 的同时,能够最大程度减少测试成本,其测试成本的计算公式如式(1)所示。

$$J=\sum_{i=1}^{m}\Big(\sum_{j=1}^{T_{f_i}}c_{T_{f_i}[j]}\Big)p(f_i) \tag{1}$$

其中,T_{f_i} 表示用于隔离故障 f_i 所用的测试序列,$|T_{f_i}|$ 表示其测试序列的长度;

2 高斯,柯西变异算子

在传统的进化算法中,假设种群大小为 N_p,第 g 代种群个体 i 表示为 $x_{i,g}=(x_1,x_2,x_3,\cdots,x_D,\sigma_1,\sigma_2,\sigma_3,\cdots,\sigma_D)$,其中 D 表示解向量的维度,其中 $\vec{x}_i=(x_1,\cdots,x_D)$ 为个体的表现向量,$\vec{\sigma}_i=(\sigma_1,\cdots,\sigma_D)$ 为表现向量的变异因子,父代经过以下变异操作产生后代。

$$\sigma_i^{g+1}(j)=\sigma_i^g(j)\exp\{\tau_1\cdot N(0,1)+\tau_2\cdot N_j(0,1)\} \tag{2}$$

$$x_i^{g+1}(j)=x_i^g(j)+\sigma_i^{g+1}(j)\cdot\varphi^g(j),\quad i=1,2,\cdots,N_p,j=1,2,\cdots,D \tag{3}$$

其中,$N(0,1)$ 表示个体 i 的随机数,$N_j(0,1)$ 为个体 i 的第 j 维向量的随机数,两者均满足标准正态分布;参数 $\tau_1=\dfrac{1}{\sqrt{2N_p}},\tau_2=\dfrac{1}{\sqrt{2\sqrt{N_p}}},\sigma_i^0(j)=3\sqrt{D}$[14] 在式(3)中,当 $\varphi^g(j)$ 满足不同分布的随机数时,得到的变异算子也就不一样;

$\varphi^g(j)$ 在式(3)中可以理解为步长;当 $\varphi^g(j)$ 满足高斯分布时,式(3)被拓展为高斯变异算子;当 $\varphi^g(j)$ 满足柯西分布时,式(3)被拓展为柯西变异算子;一维标准的高斯分布和柯西分布如图 1 所示。

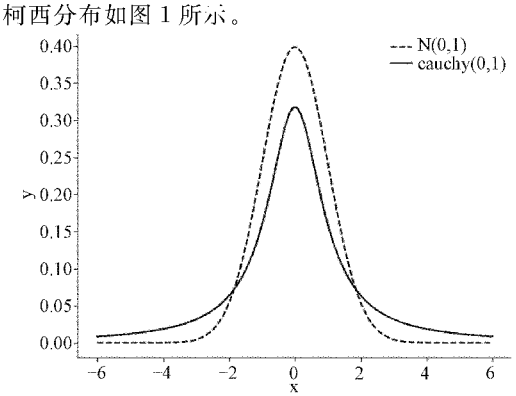

图 1 高斯-柯西一维分布图

从图中可以看出，相比高斯分布，柯西分布的两端比较宽，其变异步长会较大，因此柯西变异算子可以帮助算法更容易跳出当前位置，保证算法的全局探索能力；高斯分布更为集中，步长较小，有利于增强算法的局部寻优，保证算法的局部开发能力。

3　改进的差分进化算法

3.1　算法的基本思想

根据“No Free Lunch”定理[15]可以知道：没有任何一种算法能够解决包含不同特征的多个问题，DE 算法采用的是实数编码的方式，其算法流程包括：初始化种群、变异、交叉和选择。DE 算法的特别之处在于变异操作阶段使用的差分策略，即通过选取种群中不同的个体之间的差分向量对当前个体进行扰动，以达到变异的目的。根据差分策略的作用可以分为两类，即具有增强种群多样性的差分策略或增强局部开发特性的差分策略；在本文中为两者构建“探索策略池”和“开发策略池”。为了避免 DE 算法陷入局部最优解，从而引入高斯，柯西变异算子与策略池进行组合，以满足种群中每个个体在不同时期，具有不同的进化策略，进一步提高 DE 算法的鲁棒性和高效性。

3.2　算法的简要描述

初始化种群 $P_{op}=[x_1,x_2,x_3,x_4,\cdots,x_{Np}]$，其中 N_p 表示为种群的规模，种群个体 i 表示为 $x_i=[x_{i,1},x_{i,2},x_{i,3},x_{i,4},\cdots,x_{i,D}]$，对于最小化目标函数 $\min f(x_i)$，x_i 表示为目标函数的解。其中 $x_{i,j}^{Low}\leqslant x_{i,j}\leqslant x_{i,j}^{Up}$，$j\in[1,D]$，$D$ 表示解向量的维度。

变异操作：会直接影响整个算法的性能，第 g 代个体 $\{x_{i,g}\mid i=1,2,\cdots,N_p\}$ 通过下列差分策略产生试验向量 $v_{i,g}$：

1) DE/rand/1

$$v_{i,g}=x_{r0,g}+F_{i,g}(x_{r1,g}-x_{r2,g}) \tag{4}$$

2) DE/rand/2

$$v_{i,g}=x_{r0,g}+F_{i,g}(x_{r1,g}+x_{r2,g}-x_{r3,g}-x_{r4,g}) \tag{5}$$

3) DE/current-to-rand

$$v_{i,g}=x_{i,g}+\lambda(x_{r0,g}-x_{i,g})+F_{i,g}(x_{r1,g}-x_{r2,g}) \tag{6}$$

4) DE/best/1

$$v_{i,g}=x_{best,g}+F_{i,g}(x_{r0,g}-x_{r1,g}) \tag{7}$$

5) DE/rand-to-best/1

$$v_{i,g}=x_{r0,g}+\lambda(x_{best,g}-x_{r0,g})+F(x_{r1,g}-x_{r2,g}) \tag{8}$$

6) DE/current-to-best/1

$$v_{i,g}=x_{i,g}+\lambda(x_{best,g}-x_{i,g})+F(x_{r1,g}-x_{r2,g}) \tag{9}$$

其中，参数 $r0,r1,r2,r3,r4$ 是从集合 $\{1,2,3,\cdots,N_p\}$ 中选择的，满足条件：互不相等且均不与 i 相等；λ 表示为 0～1 之间的随机数；$F_{i,g}$ 表示第 g 代种群中个体 i 的缩放因子。在本文中构建了两个策略池；编号 1)，2)，3) 的差分策略具有增强种群多样性的特点，为其构建“探索策略池”，编号 4)，5)，6) 的差分策略具有良好的局部开发能力以及加快算法收敛的特点，为其构建为“开发策略池”。

更新操作：为了保证解的有效性，需要对经过变异操作得到的试验向量 v_i 的每一维向量进行判断，判断其是否满足边界条件并按照式(10)进行更新。

$$v_{j,i,g}=\begin{cases}2x_{i,j}^{Low}-v_{j,i,g}, & v_{j,i,g}<x_{i,j}^{Low}\\ 2x_{i,j}^{Up}-v_{j,i,g}, & v_{j,i,g}>x_{i,j}^{Up}\end{cases} \tag{10}$$

交叉操作：对 x_i 和 v_i 进行二项式选择操作，产生试验向量 u_i，其中采用策略 3) 进行变异操作的个体 x_i 不进行交叉操作；

$$u_{j,i,g}=\begin{cases}v_{j,i,g}, & rand(0,1)\leqslant CR_i \text{ or } j=j_{rand}\\ x_{j,i,g}, & \text{其他}\end{cases} \tag{11}$$

其中，$rand(0,1)$ 表示生成[0,1]之间的随机数，j_{rand} 表示为[1,D]之间的随机整数；CR_i 表示个体 x_i 的交叉概率。

选择操作：在 x_i 和 u_i 之间进行选择，选择更好的个体进入下一代；

$$x_{i,g+1}=\begin{cases}u_{i,g}, & f(u_{i,g})<f(x_{i,g})\\ x_{i,g}, & \text{其他}\end{cases} \tag{12}$$

为了标识该个体是优良个体，定义 $flag_i$ 表示个体 i 状态，初始值为 0，其中 $flag_i$ 更新方式如下：

$$flag_{i,g+1}=\begin{cases}1, & f(u_{i,g})<f(x_{i,g})\\ 0, & \text{其他}\end{cases} \tag{13}$$

3.3　算法参数设置

定义 S_F，S_{CR} 两个变量为存储第 g 代优良个体的变异概率和交叉概率的集合。参数 u_F 和 u_{CR} 的初始值均设置为 0.5。

在第 g 代中，个体 i 的交叉因子 CR_i 由满足均值为 u_{cr}，标准差为 0.1 的标准正态分布产生，如式(14)所示。

$$CR_i=randn(u_{CR},0.1) \tag{14}$$

其中，参数 u_{CR} 在每一代结束之后会按照式(15)进行更新，$c\in(0,1)$，$mean_A(x)$ 表示为算数平均函数。

$$u_{CR}=(1-c)\cdot u_{CR}+c\cdot mean_A(S_{CR}) \tag{15}$$

类似的，在第 g 代中个体 i 的变异因子 F_i 由满足均值为 u_F，标准差为 0.1 的柯西分布产生，若 $F_i\geqslant 1$，则 $F_i=1$，若 $F_i\leqslant 0$，则重新产生。

$$F_i=randc(u_F,0.1) \tag{16}$$

其中，参数 u_F 在每一代结束之后会按照式(17)进行更新，$c\in(0,1)$，$mean_L(x)$ 表示为 $Lehmer$ 均值函数。

$$u_F=(1-c)\cdot u_F+c\cdot mean_L(S_F) \tag{17}$$

$$mean_L(S_F)=\frac{\sum_{F_i\in S_F}F_i^2}{\sum_{F_i\in S_F}F_i} \tag{18}$$

4　GCMDE 求解 OTP

4.1　实数编码测试序列

种群中的个体 x_i，其 $x_{i,j}$ 表示为个体 x_i 中第 j 个测试

点,其个体维度表示为测试点的个数。假设个体 x_i 为 $[t_3, t_1, t_2, t_4, t_5]$,在本文中将测试点映射为0～1区间的均值,如图2所示。对应的测试点在GCMDE算法中编码为实数。编码为[0.50,0.10,0.30,0.70,0.90],当求解适应度函数时,执行实数解码为测试点,即执行图2中测试点区间的反映射。

测试集	t_1	t_2	t_3	t_4	t_5
区间	[0,0.2)	[0.2,0.4)	[0.4,0.6)	[0.6,0.8)	[0.8,1.0]
实数编码	0.10	0.30	0.50	0.70	0.90

图2 测试点的映射

4.2 测试序列的诊断方式

在多值属性系统的关联性矩阵中,矩阵元素 d_{ij} 的取值大小一定程度上反应了其故障状态发生的严重程度。使用其中某一个测试点会将当前故障模糊集分为 n 个模糊子集,其中 $2 \leqslant n \leqslant k$;在GCMDE算法中,总是先诊断故障严重程度更大的故障;假设某个测试点的测试结果为[0,2,3,1,4],在GCMDE中诊断故障的优先级为 $[f_5, f_3, f_2, f_4, f_1]$。对于测试序列的诊断方式有如下两种:

方式一:当前测试点用于检测当前模糊集合中的某一个单独的模糊子集。

方式二:当前测试点用于检测当前模糊集合中的 n 个模糊子集。

此外,为保持种群中个体基因的可传递性,在GCMDE算法的整个迭代过程中,个体维度是固定不变的,在进行诊断过程中,若遇到无效测试点,即该测试点对于当前作用的模糊集没影响;直接跳过该测试点,使用下一个测试点,依此类推直到测试结束。

4.3 GCMDE解决OTP问题步骤

Step1. 初始化五元组,种群规模为 N_p,个体维度为dimension;最大迭代次数 $maxRounds$,当前迭代次数 $curRound$,允许个体 i 的最大尝试代数 $TestedGen$ 设置为1/5dimension取整,最大未更新代数 $UnupdatedGen$ 设置为 $1/20Np$ 取整。创建向量:$flag$ —存放第 g 代个体是否为优良个体标识,初始值为0;$isExplore$ —用来判断个体使用的策略池,初始值为 $False$;$changeStratery$ —用来判断更改策略池,初始值为0;$mutOperator$ —用来判断个体需要追加何种变异因子,初始值为0;

Step2 $while curRound \leqslant maxRounds$:重复以下步骤,直到算法结束;

Step2.2 $for\ i\ to\ N_p$:重复以下步骤。

Step2.2.1 变量 $isExplore$ 若为 $False$,使用"探索策略池"随机产生一个试验向量;反之使用"开发策略池"产生三个试验向量并选择最适合当前个体的试验向量;得到的试验向量为 $\boldsymbol{v}_i$。

Step2.2.2 若 $curRound \%UnupdatedGen = 0$

Step2.2.2.1 若 $mutOperator > 0$,则在 $\boldsymbol{v}_i$ 的基础上追加高斯变异算子并设置 $mutOperator = 0$;否则在 $\boldsymbol{v}_i$ 的基础上追加柯西变异算子。

Step2.2.3 对 $\boldsymbol{v}_i$ 进行更新操作。

Step2.2.4 对向量 $\boldsymbol{x}_i$ 和试验向量 $\boldsymbol{v}_i$ 进行二项式交叉生成 $\boldsymbol{u}_i$,倘若个体 $\boldsymbol{x}_i$ 使用的是策略3),则不进行交叉。

Step2.2.5 在向量 $\boldsymbol{x}_i$ 和 $\boldsymbol{u}_i$ 之间进行贪婪选择。

Step2.2.5.1. 若为优良个体,$flag_i = 1$,$changeStratery_i + 1$,$mutOperator_i + 1$。否则 $flag_i = 0$。

Step2.2.6 $i = i + 1$;

Step2.3 $curRound = curRound + 1$;

Step2.4 若 $curRound \%TestedGen = 0$;

Step2.4.1 若 $changeStratery_i > 0$,变量 $isExplore$ 设置为 $True$,并设置参数 $changeStratery_i = 0$;否则变量 $isExplore$ 仍为 $False$;

Step3 算法结束;

5 实验结果与分析

GCMDE算法解决OTP问题,是在python环境下编写程序并加以验证,实验平台配置为Intel六核2.6 GHzCPU和16 GB RAM。

5.1 多值属性系统实例

1)实例1

采用文献[6]中的多值属性系统进行计算,该系统由8个故障状态构成 F,14个测试点构成 T,该多值属性系统的维度为3。采用GCMDE算法求得的最优隔离单故障诊断结果如表2所示,可以看出GCMDE算法可以得到和改进信息熵算法一样的结果,但是生成的故障诊断树的节点数目远小于改进的信息熵算法[16]。生成节点数目与算法的时间复杂度以及空间复杂度成正比,GCMDE可节省更多的存储空间并且耗时更短,算法更为高效。

表2 实例1的单故障测试序列

故障状态	GCMDE测试序列	改进信息熵测试序列
f_1	$t_1t_4t_{12}$	$t_1t_4t_{12}$
f_2	t_1t_2	t_1t_2
f_3	t_1t_2	t_1t_2
f_4	t_1t_4	t_1t_4
f_5	$t_1t_4t_{12}$	$t_1t_4t_{12}$
f_6	$t_1t_4t_{12}t_{10}$	$t_1t_4t_{12}t_{10}$
f_7	$t_1t_4t_{12}t_{10}t_{11}$	$t_1t_4t_{12}t_{10}t_{11}$
f_8	$t_1t_4t_{12}t_{10}t_{11}$	$t_1t_4t_{12}t_{10}t_{11}$
代价	14.91	14.91
节点数目	14	152

2)实例 2

采用文献[9]中的多值属性系统为例，如表 2 所示，该系统有 20 个故障状态，8 个测试点，该多值属性系统的维度为 10；

当种群大小设置为 100，个体维度设置为 15，最大迭代次数设置为 100，GCMDE 算法单独运行 30 次，平均迭代代数为 6 代，最优结果为 1.516。这与 ANT-TS[9] 和 Rollout[7] 算法得到的结果一致。得到的最优测试序列为 $[t_5,t_6,t_4,t_7,t_1,t_3,t_1,t_6,t_8,t_2]$，按照诊断方式一得到的故障诊断树如图 3 所示。

5.2　二值属性系统

二值属性系统的关联性矩阵中，d_{ij} 的取值为 0 或者 1，其代表的物理含义为第 j 个测试点测试第 i 个故障时，没有发生故障或者是可能发生故障。为了进一步说明 GCMDE 算法具有较好的性能，使用 IVDE[10] 中构建的包含故障检测率，故障隔离率以及测试点数目的适应度函数，通过下列两个二值属性系统为例进行实验对比分析。

1)反坦克系统

该系统由 13 个故障状态构成 F，12 个测试点构成 T，具体的故障状态以及测试成本等如文献[17]所示，GCMDE 独立运行 30 次，得到的最佳解决方案如表 3 所示。通过表 3 可以看出，与自适应测试离散粒子群(self-adaptive test discrete binary PSO，SADPSO)[17]算法和惯性速度差分进化(IVDE)算法相比较，GCMDE 相较于 SADPSO 和 IVDE，除了在满故障隔离率的指标下得到相同的测试点数目和成本，在其他隔离率指标下均得到了比较少的测试点数目，而且成本也可以较大程度上减少；因此可以看出 GCMDE 更有效。

表 3　实例 2 多值属性系统

故障状态	测试点 t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	概率
	测试费用								
	1	1	1	1	1	1	1	1	
f_1	4	2	3	1	4	3	2	2	0.700
f_2	7	6	6	4	8	6	5	3	0.004
f_3	4	2	3	1	2	3	1	2	0.001
f_4	1	0	0	0	1	1	0	0	0.005
f_5	8	6	6	4	9	6	5	3	0.025
f_6	5	4	6	4	7	6	5	3	0.002
f_7	6	6	6	4	8	6	5	3	0.004
f_8	2	0	0	0	0	0	0	0	0.005
f_9	0	0	0	0	1	1	0	0	0.001
f_{10}	4	2	1	1	2	3	1	2	0.029
f_{11}	7	5	6	4	8	5	5	3	0.028
f_{12}	4	2	2	1	3	3	2	2	0.003
f_{13}	7	3	5	3	6	4	5	3	0.028
f_{14}	4	2	4	1	5	3	3	2	0.003
f_{15}	3	0	0	0	1	1	0	0	0.027
f_{16}	1	1	0	0	1	1	0	0	0.027
f_{17}	7	6	6	2	8	6	5	3	0.027
f_{18}	1	0	0	0	1	2	0	0	0.027
f_{19}	7	6	6	4	8	6	4	3	0.027
f_{20}	1	0	0	0	1	1	0	1	0.027

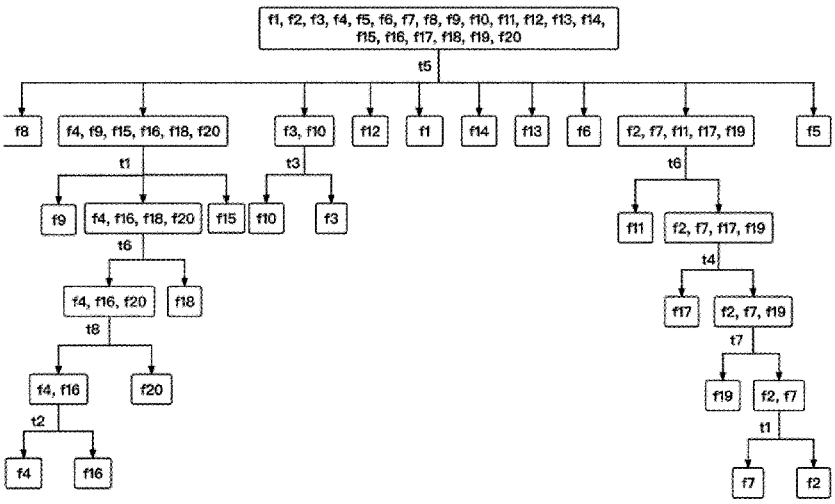

图 3　故障诊断树

独立运行 30 次，得到的最佳解决方案如表 4 所示。

通过表 4 可以看出，当 FIR 取 90%，70%的时候，得到的测试点数目与 IVDE 相同，但是所需的测试成本均较少，当 $FIR \geqslant 60\%,80\%$ 时所得到的测试点数目以及测试成本均表现较好，当 $FIR \geqslant 80\%$ 时，仅用 10 个测试点就能

表 4　反坦克系统在指定隔离率的优化结果比较

算法	FIR 目标	≥95%	≥90%	≥85%	≥80%	≥75%	≥70%
SADPSO	FIR	100%	94%	89%	83%	79%	73%
	测试点数目	11	10	10	9	9	8
	成本	4.86	4.63	4.56	4.32	4.14	3.9
IVDE	FIR	100%	94%	89%	83%	79%	73%
	测试点数目	11	10	10	9	9	8
	成本	4.75	4.52	4.44	4.21	4.02	3.9
GCMDE	FIR	100%	92%	85%	81%	75%	70%
	测试点数目	11	9	9	8	8	7
	成本	4.75	4.29	3.74	3.52	3.16	2.80

表 5　超外差接收机系统在指定隔离率的优化结果比较

算法	FIR 目标	≥90%	≥80%	≥70%	≥60%
SADPSO	FIR	94.65%	80.26%	74.69%	60.26%
	测试点数目	9	13	10	10
	成本	3.26	3.13	3.04	2.68
IVDE	FIR	93.53%	80.02%	74.85%	60.04%
	测试点数目	6	13	4	9
	成本	3.25	3.16	2.84	2.68
GCMDE	FIR	93.42%	80.09%	74.92%	60.12%
	测试点数目	6	10	4	8
	成本	2.99	2.34	2.06	1.69

够满足条件,其最优测试序列为:[t_{28},t_{34},t_8,t_{26},t_{21},t_{19},t_1,t_{27},t_{29},t_5],其测试序列通过诊断方式二得到的故障树如图 4 所示。

5.3　两种诊断方式的优缺点

以文中的超外差接收机系统实例,当 $FIR = 100\%$ 时,设置种群大小为 100,最大迭代次数为 1 000,对于两种测试序列的诊断方式,GCMDE 分别运行 30 次,方式一收敛到最优解的平均迭代次数为 78 代,得到的最优测试序列为:[t_{34},t_8,t_{19},t_{26},t_{21},t_{30},t_{30},t_{30},t_{28},t_{29},t_{32},t_{31},t_{14},t_{26},t_{31},t_7,t_5,t_{14},t_{21},t_{22},t_{10}]测试序列长度为 21,其测试代价为 3.349 79;方式二收敛到最优解的平均迭代次数为 46,得到的最优测试序列为[t_{34},t_8,t_{19},t_{30},t_{28},t_{26},t_{29},t_{32},t_{31},

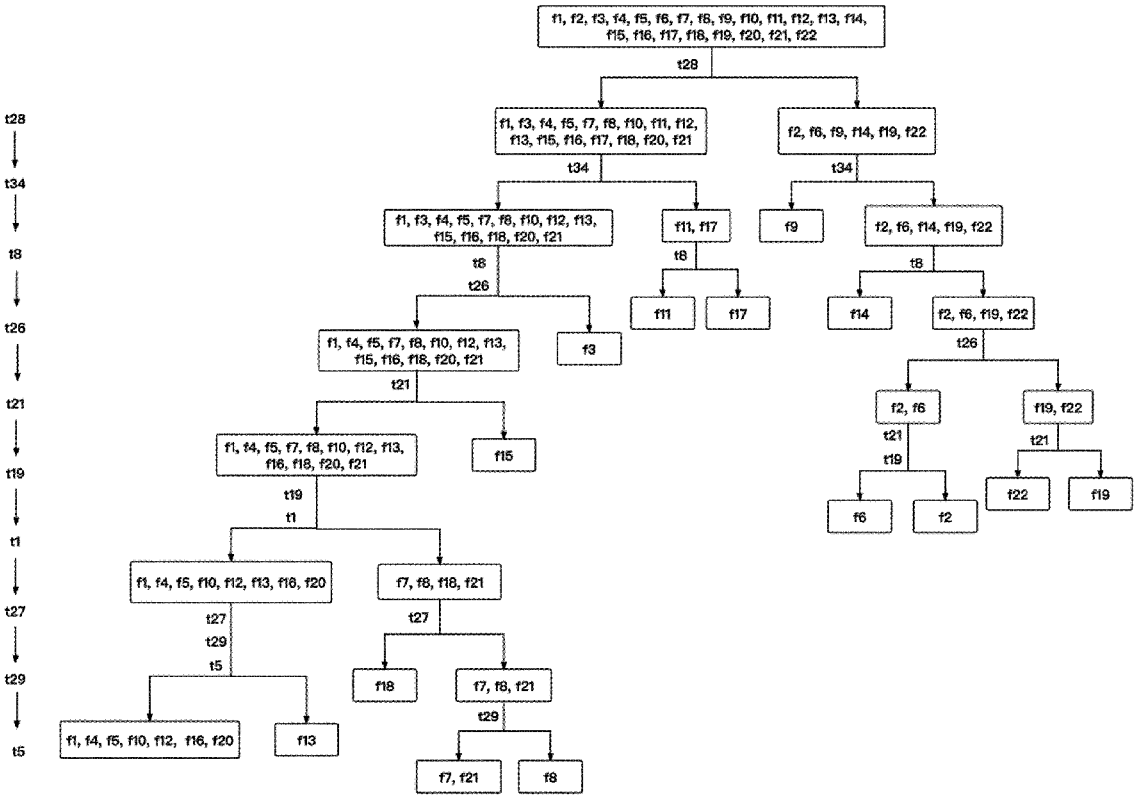

图 4　故障诊断树($FIR \geq 80\%$)

$t_7,t_5,t_{21},t_{22},t_{10},t_{14}]$，测试序列长度为 15，其测试代价亦为 3.349 79。但是当用同样的方式处理实例 2 的多值属性矩阵时，方式一得到的测试序列为：$[t_5,t_6,t_4,t_7,t_1,t_3,t_1,t_6,t_8,t_2]$，然而方式二却得不到与之对应的最优测试序列；通过以上实例分析，可以得到以下结论：

1)通过方式一得到的测试序列长度较长，会导致算法收敛的较慢，时间复杂度较大；但是得到的最优解较好。

2)方式二得到的测试序列长度和时间复杂度都较优，但是得到的最优解不一定比方式一得到的最优解好，甚至会出现得不到最优解的情况。

3)在处理小型 D 矩阵时，方式一和二都可以，但是更偏向于方式一，反之更偏向于方式二。

6　结　　论

本文针对多值属性系统的最优测试序列问题，首先分析了多值属性系统的关联性矩阵中元素的实际物理意义，并提出基于高斯-柯西变异算子的多差分策略差分进化算法来优化该问题。通过两种变异算子与两种不同特征的策略池共同作用，并给出具体的算法步骤，可以避免算法陷入局部最优解，进而得到最优结果。通过不同的实例对比剪枝算法，改进的信息熵算法，离散粒子群算法以及具有惯性速度的差分进化算法，实验结果表明，GCMDE 都具有一定的优势，具有实际应用价值。文章最后对于两种测试序列的诊断方式，给出了一些结论，具有一定的可参考性。但是并未深入，有待进一步的研究。

参考文献

[1] 康守强，刘哲，王玉静，等. 基于改进 DQN 网络的滚动轴承故障诊断方法[J]. 仪器仪表学报，2021，42(3)：201-212.

[2] 赵宁. 基于神经网络的三相整流电路故障诊断策略研究[J]. 国外电子测量技术，2021，316(3)：88-93.

[3] PATTIPATIK R，ALEXANDRIDIS M G. Application of heuristic search and information theory to sequential fault diagnosis[J]. IEEE Transactions on Systems Man & Cybernetics，1990，20(4)：872-887.

[4] 韩露，史贤俊，林云，等. 测试不可靠条件下基于精华蚂蚁系统的诊断策略优化方法[J]. 电子测量与仪器学报，2021，243(3)：130-136.

[5] 潘兴涛，赵文俊. 基于霍夫曼编码的与或树启发式的多值测试算法研究[J]. 科学技术创新，2017(33)：15-16.

[6] 黄以锋，景博，茹常剑. 基于信息熵的多值属性系统诊断策略优化方法[J]. 仪器仪表学报，2011，32(5)：1003-1008.

[7] 黄以锋，景博. 基于 Rollout 算法的多值属性系统诊断策略[J]. 控制与决策，2011，26(8)：1269-1272.

[8] TIAN H，DUAN F，FAN L，et al. Fault diagnostic strategy of multivalued attribute system based on growing algorithm[J]. Proceedings of the Institution of Mechanical Engineers，Part O：Journal of Risk and Reliability，2019，233(2)：235-245.

[9] 田恒，张文虎，邓四二，等. 基于改进蚁群算法的多值属性系统故障诊断策略[J]. 控制与决策，2021，36(11)：2722-2728.

[10] 邱晓红，李渤，李靖. 满足故障隔离率指标的测试序列优化差分进化算法[J]. 数据采集与处理，2016，31(6)：1132-1140.

[11] WANG Y，CAI Z，ZHANG Q. Differential evolution withcomposite trial vector generation strategies andcontrol parameters[J]. IEEE Transactions on Evolutionary Computation，2011，15(1)：55-66.

[12] 向万里，马寿峰，安美清. 具有 Pbest 引导机制的适应性多策略差分进化算法[J]. 模式识别与人工智能，2013，26(8)：711-721.

[13] WU G，SHEN X，LI H，et al. Ensemble of differential evolution variants[J]. Information Sciences，2018，423：172-186.

[14] LEE C Y，YAO X. Evolutionary programming using mutations based on the Lévy probability distribution[J]. IEEE Transactions on Evolutionary Computation，2004，8(1)：1-13.

[15] WOLPERT D H，MACREADY W G. No free lunch theorems for optimization[J]. Evolutionary Computation，IEEE Transactions on，1997，1(1)：67-82.

[16] 刘珊珊，吕超. 改进信息熵算法的最优测试序列生成方法[J]. 电子测量技术，2013，(12)：28-31.

[17] YANG C，YAN J，LONG B，et al. A novel test optimizing algorithm for sequential fault diagnosis[J]. Microelectronics Journal，2014，45(6)：719-727.

作者简介

邱晓红，教授，主要研究方向为智能计算和系统仿真。

E-mail：jxauqiu@163.com

徐聪(通信作者)，硕士，主要研究方向为多值属性系统诊断策略研究。

E-mail：2499982577@qq.com