软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2018,29(9):2632−2648 [doi: 10.13328/j.cnki.jos.005399]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

基于混合智能优化算法的复杂软件可靠性分配*

徐 悦, 皮德常

(南京航空航天大学 计算机科学与技术学院,江苏 南京 211106)

通讯作者: 皮德常, E-mail: dc.pi@nuaa.edu.cn

摘 要: 软件可靠性是系统设计、研究和运行过程中必须考虑的关键因素之一.与目前大多数软件可靠性分配的研究主要局限于简单的串并联模型不同,是将最优化算法应用于大型复杂软件系统的可靠性分配.针对分布估计算法收敛速度快、全局搜索能力强但易于陷入局部最优,且差分进化算法局部搜索能力强,但搜索速度略慢的问题,提出一种元启发式算法——基于罚函数的混合分布估计和自适应交叉差分进化的优化算法(PHEDA-SCDE),该算法收敛速度快、全局搜索能力强且不易陷入局部最优.基于 4 种特定的体系结构风格——顺序、并发、循环、容错,对复杂软件可靠性进行评估.为不失算法通用性,采用 3 个仿真算例进行实验,分别为单输入单输出系统、单输入多输出系统和多输入多输出系统.实验结果表明,PHEDA-SCDE 算法在软件可靠性分配方面与同类算法相比,具有明显的可行性和有效性.

关键词: 软件可靠性分配;软件可靠性评估;分布估计算法;差分进化算法;可靠性优化

中图法分类号: TP311

中文引用格式:徐悦,皮德常.基于混合智能优化算法的复杂软件可靠性分配.软件学报,2018,29(9):2632−2648. http://www.jos.org.cn/1000-9825/5399.htm

英文引用格式: Xu Y, Pi DC. Complex software reliability allocation based on hybrid intelligent optimization algorithm. Ruan Jian Xue Bao/Journal of Software, 2018,29(9):2632−2648 (in Chinese). http://www.jos.org.cn/1000-9825/5399.htm

Complex Software Reliability Allocation Based on Hybrid Intelligent Optimization Algorithm

XU Yue, PI De-Chang

(School of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China)

Abstract: Software reliability problem is one of the key factors in the process of system design, research and running. Different from most current researches on software reliability allocation limited to series parallel models, an effective optimization algorithm is applied to large complex software reliability allocation in this paper. Estimation of distribution algorithm (EDA) has fast convergence rate and strong global search capability, but is easily trapped in local optimization. Differential evolution (DE) has good local search capability with slower convergence speed. To address the issue, a new penalty guided hybrid estimation of distribution and self-adaptive crossover differential evolution algorithm (PHEDA-SCDE) is proposed in this paper. PHEDA-SCDE has fast convergence rate and strong global search capability. Also, it is not easily trapped in local optimization. In addition, software reliability is estimated based on four specific architecture styles—sequential, parallel, circulation and fault tolerant. To demonstrate the generality of the algorithm, experiments are carried out on three numerical examples including single-input/single-output system, single-input/multiple-output system and multiple-input/multiple-output system. The experimental results show that the PHEDA-SCDE is significantly feasible and efficient in reliability allocation compared with similar algorithms.

* 基金项目: 国家自然科学基金(U1433116); 中央高校基本科研业务费专项资金资助(NP2017208)

Foundation item: National Natural Science Foundation of China (U1433116); Fundamental Research Funds for the Central Universities (NP2017208)

本文由演化学习专题特约编辑俞扬副教授、钱超副研究员推荐.

收稿时间: 2017-06-28; 修改时间: 2017-08-22; 采用时间: 2017-09-26; jos 在线出版时间: 2017-11-13

CNKI 网络优先出版: 2017-11-13 14:13:22, http://kns.cnki.net/kcms/detail/11.2560.TP.20171113.1413.006.html

Key words: software reliability allocation; software reliability evaluation; estimation of distribution; differential evolution; reliability optimization

多年开展软件可靠性的研究表明,可靠性设计对软件系统可靠性有重要影响.要提高软件系统的可靠性,关键在于系统可靠性设计.软件系统可靠性的预计和分配是大型复杂软件系统可靠性设计的主要内容之一,也是可靠性设计的基础[1].可靠性预计和分配处于软件可靠性设计的前期,并贯穿软件设计的始终.显然,软件可靠性预计和分配对整个软件系统可靠性的重要性是不言而喻的.

软件可靠性分配是将系统可靠度指标合理地分配给组成该系统的各个构件,确定构件可靠性定量要求,从而保证整个系统的可靠性指标.可靠性分配是软件可靠性设计中必不可少的一个环节.软件可靠性分配技术主要分为两类:无约束的可靠性分配和有约束的可靠性分配.前者是不考虑成本等非可靠性因素,将可靠性按照权重大小分配给各个构件.后者是指采用优化技术进行可靠性分配,主要分为 3 类:可靠性分配、冗余分配和可靠性-冗余分配[1−3].本文侧重研究有约束的可靠性分配,主要内容是以系统可靠性为约束,使系统成本达到最小.一般来说,软件可靠性分配优化问题由公式(1)表示:

$$\begin{aligned} &\text{Min} \quad cost \\ &\text{s.t.} \quad RE \geqslant RE_0, \\ &\qquad l_i < r_i < u_i. \end{aligned} \tag{1}$$

其中,$cost$ 为软件系统总成本,r_i 是第 i 个构件的可靠性,l_i 和 u_i 是第 i 个构件可靠性的下限和上限,RE 是软件系统的可靠性,RE_0 是系统要达到的预定可靠性值.r_i 为可靠性分配的自变量,是本文需要求解优化的参数.

软件可靠性分配问题一直是当今软件工程领域的热门研究话题,许多学者在此领域进行了大量研究.Kuo 和 Prasad[1]总结了自从 1977 年以来应用于可靠性最优化的方法,主要研究了 4 类系统结构.Rocco[4]提出了一种新的进化算法,用来解决多状态双端网络的多目标优化问题.2012 年,Hsieh 和 Yeh[5]提出一种基于罚函数的人工蜂群算法,用以解决冗余分配问题(RAP).Khalili-Damghani 等人[6]对二状态多目标可靠性冗余分配问题提出一种动态自适应的粒子群优化算法,同时,他们用罚函数和修改策略来处理限制条件.为了解决时间-成本-质量的权衡问题,Tran 等人[7]提出了一种基于混合人工蜂群算法和进化计算的多目标进化算法.Zhang 和 Lei[8]根据遗传算法和粒子群的优势,提出了一种用来解决网构软件可靠性的动态混合分布算法.由于大多数现有的可靠性冗余分配模型都基于构件固定可靠性的假设,2016 年,Zhang 和 Chen[9]提出了一种新的基于粒子群优化算法.

软件可靠性评估主要有 3 种方式:基于状态的模型、基于路径的模型和附加模型[10].文献[11]根据不同的体系结构风格,评估软件可靠性.本文拟在此基础上新增一类体系结构风格——循环.Cheung[12]基于每个构件的可靠性和构件间的转移概率提出了一个马尔可夫模型用来评估软件可靠性,同时对构件进行敏感度分析.Zhang 和 Lei[8]基于马尔可夫链研究了关于网构软件可靠性计算模型.本文拟给出 4 种特定的体系结构风格,并对各自的可靠性在第 3 节进行具体分析.

本文第 1 节主要阐述软件可靠性分配问题的最优化目标模型,介绍两种子优化算法——分布估计算法和差分进化算法.在此基础上,本文提出一种新的基于罚函数的混合分布估计和自适应交叉差分进化算法,用来解决软件可靠性分配问题.第 2 节详细介绍 4 种特定的体系结构风格,并对各自的可靠性进行分析.第 3 节主要介绍算例分析,从收敛速度以及精度两个方面,采用 5 种算法进行比较和分析.最后总结全文,并对未来值得关注的研究方向进行初步探讨.

1 软件可靠性分配算法

1.1 最优化目标模型

软件可靠性分配是将系统可靠度指标合理分配给组成该系统的各个构件,确定各构件的可靠性定量要求,从而保证整个系统的可靠性指标.Henlander 等人[13]提出了两种设计可靠性和成本的方法:限制可靠性最小化成本(RCCM)和限制预算最大化可靠性(BCRM).本文利用进化计算的方法,以系统可靠性为约束,在系统成本最小

化的条件下计算各个构件可靠性,从而帮助系统决策者合理分配资源给各个构件.在此基础上,软件可靠性分配问题简化为基于费用函数的最优化目标模型,目标函数可以写成各个构件可靠性的函数:

$$\begin{aligned} \text{Min} \quad & cost=\sum_{i=1}^{n} cost(r_i) \\ \text{s.t.} \quad & RE \geqslant RE_0, \\ & 0.5<r_i<1, \\ & RE_0=0.9. \end{aligned} \tag{2}$$

其中,$cost$ 为软件系统总成本,r_i 是第 i 个构件的可靠性,RE 是软件系统的可靠性,RE_0 是系统要达到的预定可靠性值.公式(2)使得系统预计可靠性达到 0.9,每个构件可靠性在 0.5 到 1 之间的前提下满足成本最小.将约束方程作为罚函数项[14]加入到目标函数中,变成无约束的优化问题,即:

$$\text{Min} \quad cost+penalty=\sum_{i=1}^{n} cost(r_i)+k\cdot\min\{0,RE-RE_0\} \tag{3}$$

其中,k 为一充分大的负数.综上可知:如果不满足系统最低达到的可靠性值,罚函数反比例于 $RE–RE_0$.

一般在成本最小为优化目标的模型中,成本与各个构件可靠性之间的函数关系是非常重要的.Henlander 等人[13]提出了基于对失效率和成本之间关系现实假设的 3 种成本函数:线性形式、对数指数形式、逆幂形式.Mettas 等人[15]提出了基于经验和(或者)数据的可靠性成本关系函数如公式(4)所示:

$$c_i(R_i;f_i,R_{i,\min},R_{i,\max})=\mathrm{e}^{(1-f_i)\frac{R_i-R_{i,\min}}{R_{i,\max}-R_i}} \tag{4}$$

其中,c_i 表示构件 i 的成本;f_i 表示增加一个构件可靠性的可靠性,值在 0 到 1 之间;R_i 为构件 i 初始(当前)的可靠性值;$R_{i,\min}$ 为构件 i 初始(当前)的最小可靠性值;$R_{i,\max}$ 为构件 i 初始(当前)的最大可靠性值.本文采用 Mettsa 提出的指数形式函数,因为其函数满足如下几点可靠性与成本之间关系的需求.

(1) 函数值是正数;

(2) 成本是关于构件可靠性的单调增函数;

(3) 可靠性越高,成本越高;反之亦然.

1.2 子优化算法

1.2.1 分布估计算法

分布估计算法(estimation of distribution algorithm,简称 EDA)是一种新型的启发式算法[16],EDA 算法是遗传算法的改进,但与传统遗传算法不同的是,该算法没有交叉、变异等遗传操作.分布估计算法构建描述解空间的概率模型,根据目标函数选取优秀的个体集合,采用统计学习方法构造一个描述当前优秀个体解集的概率模型.对概率模型随机采样,从而得到新的种群.在此过程中,概率模型会被不断朝最优解的方向进化.

EDA 算法基本步骤如下.

步骤 1.初始化群体.

步骤 2.适应度值评价.

步骤 3.适应度值按照从高到低排序,用截断选取的策略选取 k 个优秀个体.

步骤 4.分别计算当前种群的均值和方差.

步骤 5.构建概率分布模型.

步骤 6.根据概率模型采样产生下一代.

步骤 7.若终止条件满足,则算法终止;否则转向步骤 2 继续执行.

分布估计算法的优势在于全局搜索能力强、收敛速度快,可以很快缩小搜索空间.但局部搜索能力相对较弱,容易陷入局部最优.

1.2.2 自适应交叉差分进化算法

差分进化算法(differential evolution,简称 DE)是一种基于向量的适者生存、优胜劣汰算法[17],该算法将种群

中的个体看成当前向量,随机产生自适应变异.差分进化算法根据公式(5)初始化种群为 NP 个 D 维向量,$x_i=\{x_{i,1}, x_{i,2},\ldots,x_{i,D}\}$,$i=1,2,\ldots,NP$.个体采用实数编码,公式中,$u$ 为个体上限值,l 为个体下限值:

$$x=rand(NP,D)\times(u-l)+l \tag{5}$$

DE 随机选取两个不同的向量相减产生差分向量,然后将差分向量赋予权值后与另一随机向量相加,从而生成变异个体,如公式(6)所示:

$$v_i = x_{r_1} + F\cdot(x_{r_2} - x_{r_3}) \tag{6}$$

其中,x_i 为当前目标向量,v_i 为变异产生的种群个体向量,r_1,r_2,r_3 互不相等.F 是自适应变异算子,由公式(7)生成:

$$F = F_0\cdot 2^{\lambda}, \lambda = \mathrm{e}^{1-\frac{\max gen}{\max gen+1-gen}} \tag{7}$$

其中,F_0 是初始变异算子,maxgen 表示最大进化代数,gen 表示当前进化代数.然后,将变异向量与当前目标向量随机交叉产生实验个体向量 u_i.最后,根据公式(8)比较目标向量和实验向量的适应度值,选择适应度值较优的群体向量:

$$x_i = \begin{cases} u_i, \text{ if } f(u_i) < f(x_i) \\ x_i, \text{ otherwise} \end{cases} \tag{8}$$

传统 DE 算法全局搜索能力较差,收敛速度较慢.为了改进算法的性能,自适应策略已普遍应用于差分进化算法.Qin 等人[18]提出一种自适应差分进化算法(SaDE),对实验向量和相应控制参数自适应改进.Wang 等人[19]提出一种带有改进变异策略的自适应差分进化算法(IMSaDE),包括精英备选策略(elite archive strategy)和控制参数适应策略(control parameters adaptation strategy).为解决多文档文摘(multi-document summarization)优化问题,Alguliev 等人[20]提出一种新的基于自适应变异和交叉的算法(DESAME).本文采取自适应交叉策略改进算法,旨在加强全局搜索能力,迅速得到最优解空间,如图 1 所示.

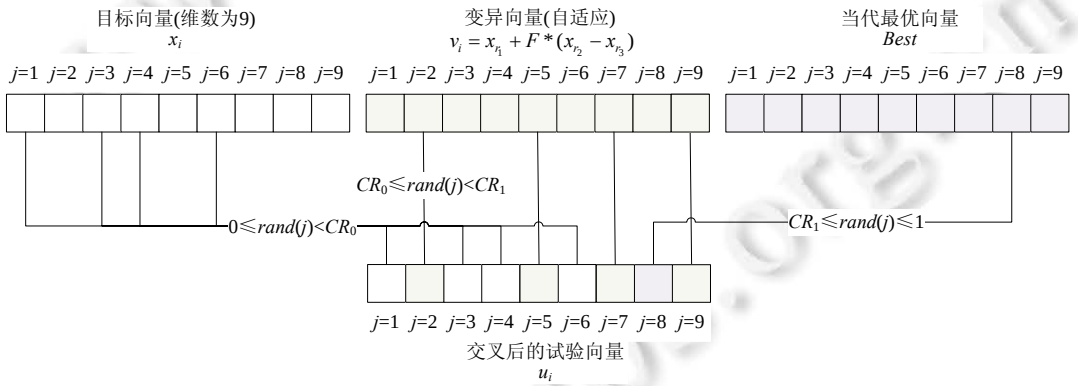


Fig.1 The self-adapative crossover of differential evolution

图 1 差分进化算法自适应交叉示意图

在交叉部分,目标向量不仅与变异向量进行交叉,而且与当代最优向量 $best$ 进行交叉,如公式(9)所示.此交叉过程是一个自适应的过程,同时,其变异和选择都具有自适应的性质.本文将改进后的差分进化算法称为自适应交叉差分进化算法(self-adaptive crossover differential evolution,简称 SCDE):

$$u_{i,j} = \begin{cases} x_{i,j}, & 0 \leqslant rand < CR_0 \\ v_{i,j}, & CR_0 \leqslant rand < CR_1 \\ best_j, & CR_1 \leqslant rand \leqslant 1 \end{cases} \tag{9}$$

其中,CR_0 和 CR_1 是用户定义的交叉常量,值在 0 和 1 之间.

SCDE 详细步骤如下所示.

步骤 1.初始化群体.

步骤 2.适应度值评价.

步骤 3.自适应变异.

步骤 4.自适应交叉.

步骤 5.选择更优向量产生下一代.

步骤 6.若终止条件满足,则算法终止;否则,转向步骤 2 继续执行.

差分进化算法具有很强的局部搜索能力,根据差分向量对种群中几个个体向量随机自适应变异.但与分布估计算法相比,DE 算法收敛速度较慢.所以在交叉阶段,当前向量不仅与变异向量交叉,还有一定概率和当前最优向量交叉,从而加快收敛速度.

1.3 混合优化算法

EDA 算法全局搜索能力强,收敛速度快.在种群进化初期,需要将解空间缩小到一定范围,进一步缩短算法在后期寻优时间.SCDE 算法局部搜索能力强,结构简单.在种群进化中后期,需要加强局部搜索能力,避免陷入局部最优.在此基础上,本文提出一种基于罚函数的混合分布估计和自适应交叉差分进化的优化算法(PHEDA-SCDE).PHEDA-SCDE 算法前期具有较强的全局搜索能力,收敛速度快,能迅速找到全局最优所在的解空间.该算法在其中后期不仅具有较强的局部搜索能力,而且避免对复杂优化问题陷入局部最优的困境.

1.3.1 决策因子和轮盘赌选择

在实践中,最常见的选择方法是轮盘赌选择方法和精英方法.本文通过决策因子 n 和轮盘赌选择将 EDA 和 SCDE 算法有效融合.轮盘赌选择[21]是一种基于适应度函数值的回放式随机采样方法,每个个体被选中的概率与适应度值大小成正比.适应度值越大,被选中的概率越高;反之亦然.其特点是优秀个体具有较大被选中的概率,且每个个体都有被选择的概率.在 PHEDA-SCDE 算法中,EDA 和 SCDE 各自对每一代种群进行优化,产生候选实验个体.本文对 EDA 产生的候选种群进行轮盘赌选择,选择出 n 个候选实验个体;同时也采用轮盘赌选择方式,在 SCDE 产生的候选种群中选择 $NP–n$ 个候选实验个体.两部分候选解组合成新一代种群,增加种群的多样性.当前个体既继承了上一代的种群信息,又产生了不同的机制特征信息.

固定两个算法种群比例或者随机设置比例是不可行的:固定种群比例,前期算法收敛速度慢,中后期算法局部搜索能力弱;随机设置算法比例,算法很有可能陷入局部最优,算法稳定性不高.本文采用公式(10)为混合算法决策因子 n 的计算公式:

$$n(gen+1)=n_{\min}+\left(1-\frac{gen}{\max gen}\right)\cdot(n(gen)-n_{\min}) \tag{10}$$

其中,gen 为当前迭代次数,maxgen 为最大迭代次数,$n_{\min}$ 和 $n_{\max}$ 分别为决策因子 n 的下限和上限.

图 2 为决策因子变化曲线,其中,$n_{\min}$=0.2,$n_{\max}$=0.9.

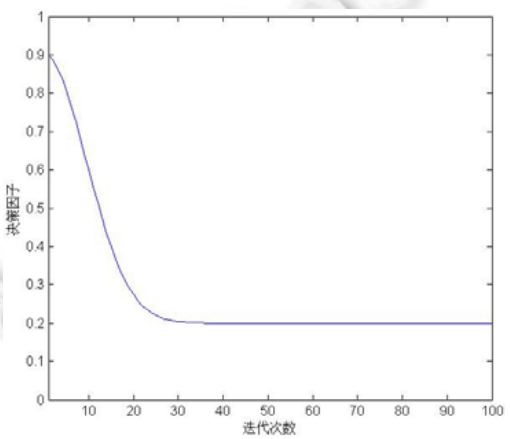


Fig.2 The variation curve of the decisive factor n

图 2 决策因子 n 变化曲线图

显然,从图 2 可以发现,决策因子随着迭代次数的增加而减少.在进化初期,选择较大数值的决策因子,EDA 算法占主导作用,这样可以将解空间快速缩小到一定范围.与此同时,一小部分个体由 SCDE 算法产生,保证了种群多样性,为算法后期寻优提供了优质解空间.随着迭代次数的增加,决策因子 n 逐渐递减,SCDE 算法在进化中后期占主导地位.EDA 算法虽然全局搜索能力强,收敛速度快,但在算法中后期很容易陷入局部最优,所以 PHEDA-SCDE 算法在其中后期通过决策因子,增大了 SCDE 算法生成个体的比例,提高算法局部搜索能力.同样的,算法仍保留部分由 EDA 算法产生的个体来提供算法的求解质量和效率.

1.3.2 PHEDA-SCDE 算法流程

PHEDA-SCDE 算法是根据父代种群的差分向量信息和概率模型分布选择适应度较高的子代,初期维持了种群的全局信息,中后期维持了种群的局部信息.PHEDA-SCDE 算法伪代码见算法 1,具体流程如图 3 所示.

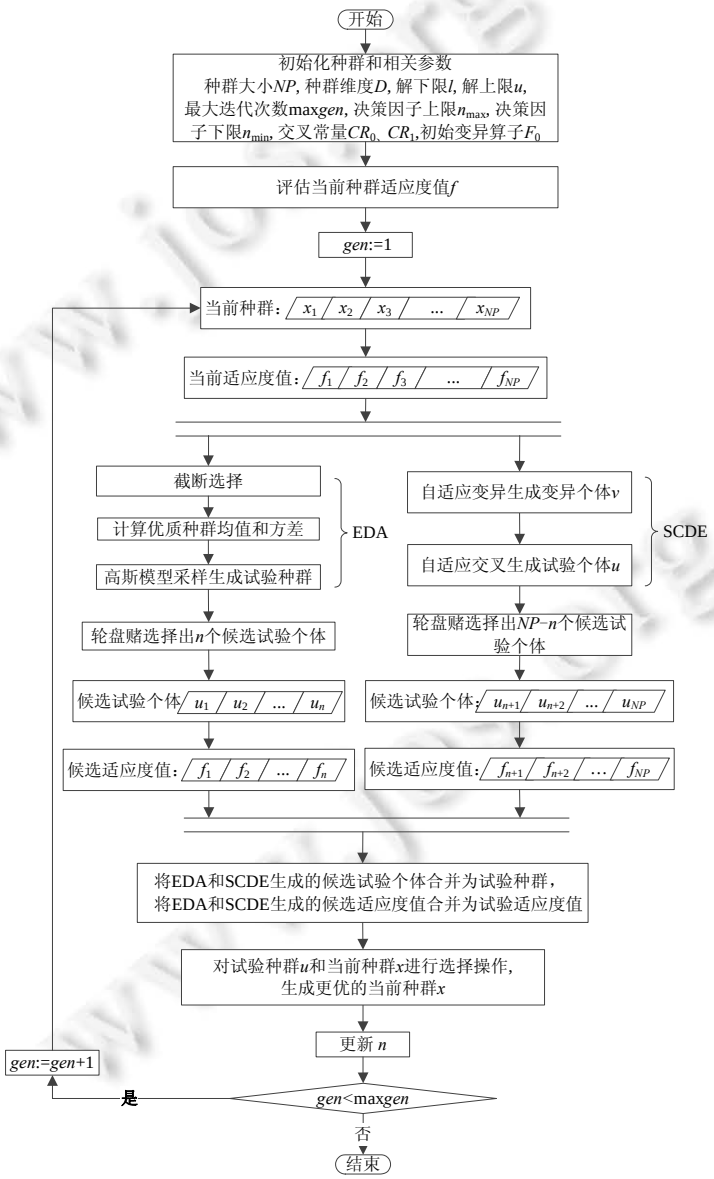


Fig.3 The procedure of PHEDA-SCDE

图 3 PHEDA-SCDE 算法流程图

算法 1. PHEDA-SCDE 算法的伪代码.

输入:种群规模 NP,种群维数 D,个体上限值 u 和个体下限值 l,决策因子最大值 $n_{\max}$ 和决策因子最小值 $n_{\min}$,最大迭代代数 maxgen,交叉算子 CR_0 和 CR_1,初始变异算子 F_0,阈值 $truncation$;

输出:种群中的最优个体及其适应度值.

开始

```
1   x:=rand(NP*D)*(u−l)+l                          //初始化种群
2   f:=fitness(x)                                  //计算适应度值
3   for gen=1 to maxgen do                         //主循环
4     [EDAx,EDAf]:=EDA(x,f,truncation)             //通过 EDA 算法对当前种群进行优化
5     [EDAu,EDAf]:=roulette(EDAx,EDAf,n)           //轮盘赌选择出 n 个候选实验个体
6     [SCDEx,SCDEf]:=SCDE(x,f,CR0,CR1,F0)          //通过 SCDE 算法对当前种群进行优化
7     [SCDEu,SCDEf]:=roulette(SCDEx,SCDEf,NP−n)    //轮盘赌选择出 NP−n 个候选实验个体
8     new:=(EDAu,SCDEu)                            //种群合并
9     newf:=(EDAf,SCDEf)                           //适应度值合并
10    if newf>f do                                 //选择个体进入下一代
11      x:=new
12      f:=newf
13    endif
14    n:=nmin+(1−gen/maxgen)*(n−nmin)              //更新决策因子
15 endfor
```

结束

本文通过一个简单算例介绍 PHEDA-SCDE 算法的进化操作,详见附录 A.

1.3.3 PHEDA-SCDE 算法收敛性分析

定理 1. PHEDA-SCDE 算法中种群进化方向是单调不增的,即 $f(X(\text{ge}n+1))\leqslant f(X(\text{ge}n))$.

证明:由公式(8)所示,PHEDA-SCDE 算法的选择算子是贪婪选择操作,总会保留更优个体,因此种群最优适应度单调不增,命题得证.

定理 2. PHEDA-SCDE 算法的种群序列$\{X(\text{ge}n),\text{ge}n\in N^+\}$为齐次 Markov 链.

证明:由于 PHEDA-SCDE 算法的种群序列:

$$X(gen+1)=T(X(gen))=((T_{crossover}\circ T_{mutation}(X(gen)))\circ T_{roulette},(T_{sample}\circ T_{truncation}(X(gen)))\circ T_{roulette})\circ T_{selection} \tag{11}$$

其中,$T_{crossover},T_{mutation},T_{roulette},T_{sample},T_{truncation},T_{selection}$ 均与 n 无关.

因此,$X(gen+1)$仅与 $X(gen)$有关,即$\{X(gen),gen\in N^+\}$是 Markov 链.显然,其转移概率大于 0,命题得证. □

定理 3. PHEDA-SCDE 算法的 Markov 链种群序列$\{X(n),n\in N^+\}$以概率 1 收敛于全局最优解集 M^*,其中,全局最优解集 $M^*=\{X^*;\forall X^*\in S,\text{有 } f(X^*)\leqslant f(X)\}$:

$$\lim_{gen\to\infty} P(X(gen)\in M^*)=1 \tag{12}$$

证明:由定理 1、定理 2 知,PHEDA-SCDE 算法为单调不增的齐次 Markov 链,则 $X(gen-1)$到 $X(gen)$的转移概率有:

$$P\{X(gen-1),X(gen)\}=\begin{cases}P\{X(gen)\in M^*\mid X(gen-1)\in M^*\}=1\\ P\{X(gen)\notin M^*\mid X(gen-1)\in M^*\}=0\\ P\{X(gen)\in M^*\mid X(gen-1)\notin M^*\}>0\\ P\{X(gen)\notin M^*\mid X(gen-1)\notin M^*\}>0\end{cases} \tag{13}$$

根据全概率公式 $P(B)=P(A)P(B|A)+P(\bar{A})P(B|\bar{A})$,$P\{X(gen)\in M^*\}$可以表示为公式(14):

$$P\{X(gen)\in M^*\}=(1-P\{X(gen-1)\in M^*\})\cdot P\{X(gen)\in M^*\mid X(gen-1)\notin M^*\}+ P\{X(gen-1)\in M^*\}\cdot P\{X(gen)\in M^*\mid X(gen-1)\in M^*\} \tag{14}$$

令 $\theta(t)=X(gen)\in M^*$, $\omega(t)=P\{X(gen)\in M^*\mid X(gen-1)\notin M^*\}$,公式(14)改写为

$$\theta(t)=(1-\theta(t-1))\omega(t)+\theta(t-1) \tag{15}$$

则:

$$\begin{aligned}1-\theta(t)=(1-\theta(t-1))\cdot(1-\omega(t))&\Rightarrow 1-\theta(t)=(1-\theta(t-2))\cdot(1-\omega(t))\cdot(1-\omega(t-1))\\&\Rightarrow ...\\&\Rightarrow 1-\theta(t)=(1-\theta(0))\cdot\prod_{k=1}^{t}(1-\omega(k))\end{aligned} \tag{16}$$

两边取极限为

$$\lim_{t\to\infty}\theta(t)=1-\lim_{t\to\infty}\left((1-\theta(0))\cdot\prod_{k=1}^{t}(1-\omega(k))\right)\Rightarrow\lim_{t\to\infty}\theta(t)=1-(1-\theta(0))\cdot\lim_{t\to\infty}\left(\prod_{k=1}^{t}(1-\omega(k))\right) \tag{17}$$

根据转移概率 $0<\omega(t)\leqslant 1$,可得 $\lim_{t\to\infty}\left(\prod_{k=1}^{t}(1-\omega(k))\right)=0$.

因此, $\lim_{t\to\infty}\theta(t)=1$,即 $\lim_{gen\to\infty}P(X(gen)\in M^*)=1$.证毕. □

2 软件可靠性评估方法

本节给出顺序、并发等 4 种特定的体系结构风格,并对各自的可靠性进行具体分析.任何复杂的大型软件系统都可以由顺序、并发、循环、容错及其扩展情况表示.

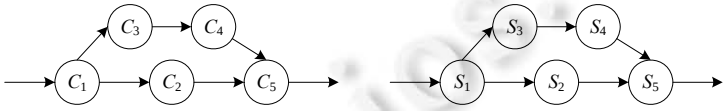
2.1 顺 序

$C1;C2$ 表示构件 $C1$ 和 $C2$ 是顺序执行,即:先执行 $C1$,再执行 $C2$.假设转移概率为 $p_{i,j}$,第 i 个状态可靠性为 $r(S_i)$,那么 $C1;C2$ 可靠性为

$$RE=r(S_1)\cdot p_{1,2}\cdot r(S_2) \tag{18}$$

假设某顺序执行的体系结构如图 4(a)所示,其中,$C_1,C_2,\ldots,C_5$ 是系统结构的构件,构件数量为 5.顺序风格的状态图 4(b)是体系结构图的一一映射,其中,$S_1,S_2,\ldots,S_5$ 是马尔可夫链的状态,状态数量为 5.则此系统可靠性为

$$RE=r(S_1)\cdot p_{1,2}\cdot r(S_2)\cdot p_{2,5}\cdot r(S_5)+r(S_1)\cdot p_{1,3}\cdot r(S_3)\cdot p_{3,4}\cdot r(S_4)\cdot p_{4,5}\cdot r(S_5) \tag{19}$$



(a) 体系结构图 (b) 状态图

Fig.4 Sequential style

图 4 顺序执行风格

2.2 并 发

$C_1\|C_2$ 表示构件 C_1 和 C_2 是并发操作.与顺序风格一次只执行一个构件不同的是:在并发风格中,两个或者多个构件可以同时执行.假设转移概率为 $p_{i,j}$,第 i 个状态的可靠性为 $r(S_i)$,其中,$p_{1,p}=p_{1,2}=p_{1,3}$,$r(S_p)=r(S_2)=r(S_3)$,那么 $C_1;(C_2\|C_3)$的可靠性为

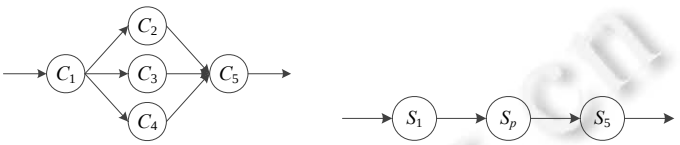
$$RE=r(S_1)\cdot p_{1,p}\cdot r(S_p) \tag{20}$$

假设某并发执行的体系结构如图 5(a)所示,其中,$C_1,C_2,\ldots,C_5$ 是系统结构的构件,构件数量为 5.特别地,构件 C_2,C_3,C_4 可以同时发生.在并发风格的状态图 5(b)中,构件 C_2,C_3,C_4 合并为并发状态 S_p,其中,S_1,S_p,S_n 是马尔可夫链的状态,状态数量为 3.假定 $p_{1,2}=p_{1,3}=p_{1,4}=p_{1,p},p_{2,5}=p_{3,5}=p_{4,5}=1$,可以得出从状态 S_p 到 S_5 的转移概率 $p_{p,5}$ 和该系

统的可靠性 RE 为

$$p_{p,5}=\prod_{i=2}^{4}p_{i,5}=1 \tag{21}$$

$$RE=r(S_1)\cdot p_{1,p}\cdot r(S_p)\cdot r(S_5) \tag{22}$$



(a) 体系结构图 (b) 状态图

Fig.5 Parallel style

图 5 并发执行风格

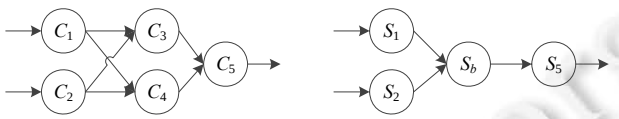
2.3 容 错

$C_1|C_2$ 表示 C_1 是主要构件,C_2 是备份构件.如果构件 C_1 故障失效,C_2 作为备份构件仍可执行相应功能,系统可正常运行.假设转移概率为 $p_{i,j}$,第 i 个状态的可靠性为 $r(S_i)$,其中,$p_{1,b}=p_{1,2}=p_{1,3}$,$r(S_b)=r(S_2)=r(S_3)$,那么 $C_1;(C_2|C_3)$ 的可靠性为

$$RE=r(S_1)\cdot p_{1,b}\cdot r(S_b) \tag{23}$$

假设某容错执行的体系结构如图 6(a)所示,其中,$C_1,C_2,\ldots,C_5$ 是系统结构的构件,构件数量为 5.特别地,构件 C_3,C_4 是容错关系,C_3 是主要构件,C_4 是备份构件.在容错风格的状态图 6(b)中,构件 C_3,C_4 合并为容错状态 S_b,其中,S_1,S_b,S_n 是马尔可夫链的状态,状态数量为 3.假定 $p_{1,3}=p_{1,4}=p_{1,b},p_{2,3}=p_{2,4}=p_{2,b},p_{3,5}=p_{4,5}=p_{b,5}=1$,那么该系统的可靠性 RE 为

$$RE=r(S_1)\cdot p_{1,b}\cdot r(S_b)\cdot r(S_5)+r(S_2)\cdot p_{2,b}\cdot r(S_b)\cdot r(S_5) \tag{24}$$



(a) 体系结构图 (b) 状态图

Fig.6 Fault tolerance style

图 6 容错执行风格

2.4 循 环

μC_1 表示循环执行构件 C_1 的相关功能,可以执行一次也可以执行多次.假设转移概率为 $p_{i,j}$,第 i 个状态的可靠性为 $r(S_i)$,其中,$p_{1,b}=p_{1,2}=p_{1,3}$,$r(S_b)=r(S_2)=r(S_3)$,那么 $C_1;(\mu C_2);C_3$ 的可靠性为

$$RE=r(S_1)\cdot p_{1,b}\cdot r(S_b) \tag{25}$$

$C_1;(\mu C_2);C_3$ 循环执行的体系结构如图 7(a)所示,其中,C_1,C_2,C_3 是系统结构的构件,构件数量为 3.特别地,构件 C_2 自身循环零次或者多次.顺序风格的状态图 7(b)也是体系结构图的一一映射,其中,S_1,S_2,S_3 是马尔可夫链的状态,状态数量为 3.

- 若系统没有执行循环操作,则系统可靠性

$$RE(0)=r(S_1)\cdot p_{1,2}\cdot r(S_2)\cdot p_{2,3}\cdot r(S_3) \tag{26}$$

- 若系统执行循环操作一次,则系统可靠性

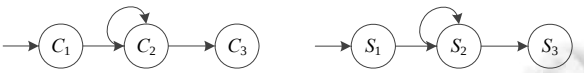
$$RE(1)=r(S_1)\cdot p_{1,2}\cdot r(S_2)\cdot p_{2,2}\cdot r(S_2)\cdot p_{2,3}\cdot r(S_3) \tag{27}$$

- 直至系统执行循环操作μ次,则系统可靠性

$$RE(\mu)=r(S_1)\cdot p_{1,2}\cdot r(S_2)\cdot (p_{2,2}\cdot r(S_2))^{\mu-1}\cdot p_{2,3}\cdot r(S_3) \tag{28}$$

所以,系统可靠性可以表示为

$$RE=\sum_{i=0}^{\mu}RE(i)=\frac{r(S_1)\cdot p_{1,2}\cdot r(S_2)\cdot p_{2,3}\cdot r(S_3)}{1-p_{2,2}\cdot r(S_2)} \tag{29}$$



(a) 体系结构图 (b) 状态图

Fig.7 Cycle style

图 7 循环执行风格

3 仿真实验结果与分析

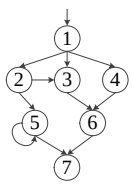
3.1 参数设置

参数设置对随机性算法的性能和效率有显著的影响[22],因此,本节通过设置参数对比分析实验,找出求解软件可靠性分配问题的较优参数.本文提出的 PHEDA-SCDE 算法包含了几个关键参数:交叉算子 CR_0 和 CR_1、初始变异算子 F_0、阈值 *truncation*、决策因子上限 $n_{\max}$ 和下限 $n_{\min}$.本文采用实验设计法(design of experiments,简称 DOE)[23],每个参数选择合理取值,将其构成算法的参数集.参数取值如下:$CR_0\in\{0.1,0.2,0.3,0.4\}$,$CR_1\in\{0.6,0.7,0.8,0.9\}$,$F_0\in\{0.2,0.4,0.6,0.8\}$,$truncation\in\{0.2,0.3,0.4\}$,$n_{\max}\in\{0,0.1,0.2\}$,$n_{\min}\in\{0.8,0.9,1\}$.这些参数对于本文提出的算法,共有 4×4×4×3×3×3=1728 种不同配置.为了保证实验结果更可靠,每一种配置都独立运行 5 次.

本节按照文献[12,24]的方法生成受控数据集进行参数设置实验,然后在测试实例集上进行验证.受控数据集详细信息见表 1.

Table 1 The control data set

表 1 受控数据集详细信息

状态图	转移概率	目标函数
	$p_{1,2}$=0.6 $p_{1,3}$=0.1 $p_{1,4}$=0.3 $p_{2,3}$=0.3 $p_{2,5}$=0.7 $p_{3,6}$=1.0 $p_{4,6}$=1.0 $p_{5,5}$=0.4 $p_{5,7}$=0.6 $p_{6,7}$=1.0	$cost+k\times\min\{0,RE-r_0\}$

为探寻参数对算法的影响,本文采用方差分析(analysis of variance,简称 ANOVA)方法来设计实验.主效应图(如图 8 所示)是显示单个参数对结果影响的显示图,根据图例可以分析得到达到哪个值的时候这个参数的影响力最大,从而找出最主要因素.从图 8 中可以明显看出:初始变异算子 F_0 对目标函数的影响最大,这主要是因为子算法 SCDE 中有 60%进行自适应交叉,当其值为 0.6 时,目标函数达到最优化;交叉算子 CR_0 与决策因子下限 $n_{\min}$ 对目标函数影响相对最小.根据图 8 分析可知,选择表 2 中所列出的参数配置较为合理.

Table 2 Parameter tuning of PHEDA-SCDE

表 2 PHEDA-SCDE 算法运行参数设置

参数类型	参数名称	值
CR_0	交叉算子	0.2
CR_1	交叉算子	0.8
F_0	初始变异算子	0.6
truncation	阈值	0.3
$n_{\max}$	决策因子上限	0.2
$n_{\min}$	决策因子下限	1.0

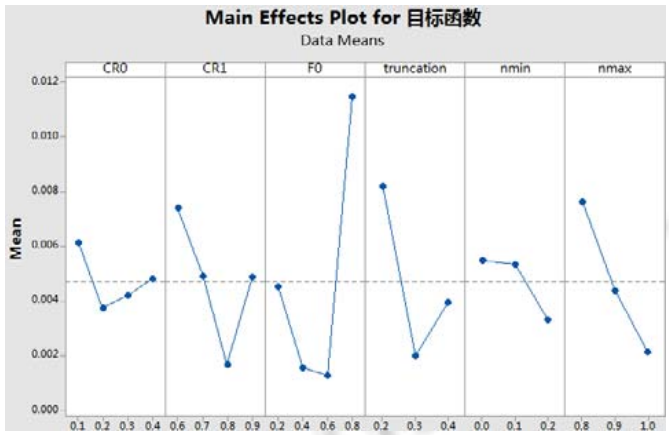


Fig.8 Main effect plot of parameters

图 8 PHEDA-SCDE 算法主效应图

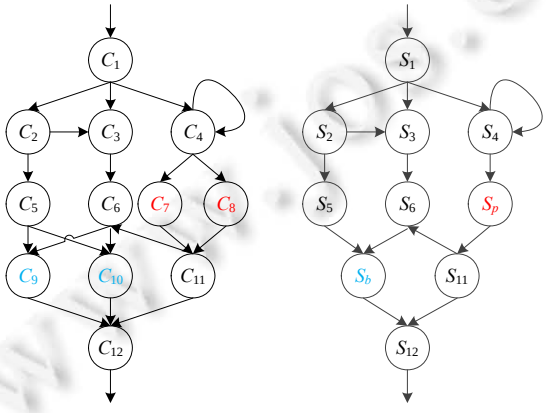
3.2 仿真算例实验与分析

3.2.1 软件可靠性分配算例

为不失算法的通用性,本文采用 3 个仿真算例,分别为单输入单输出系统、单输入多输出系统和多输入多输出系统.以某复杂软件系统可靠性优化仿真算例进行说明[8],如图 9.1 所示.算例 1 是由 12 个构件组成的单输入单输出系统,其中,C_1代表输入构件,C_{12}代表输出构件.特别地,构件 C_7和 C_8是并发关系,C_{10}是 C_{12}的容错构件,构件 C_4自身循环.系统状态如图 9.1(b)所示,是由图 9.1(a)的体系结构图转变过来.本文按照文献[24]的方法,算例 2(见图 9.2)删除算例 1 中 C_{11}和 C_{12}之间的连接、C_{11}和 C_6之间的连接,使得 C_{11}和 C_{12}均为输出构件.算例 3(见图 9.3)中系统开始于输入组件 C_1和 C_2,结束于输出组件 C_{11}和 C_{12}.

为了简化模型,本文认为,转移概率 $p_{i,j}$已经考虑了连接件的可靠性.算例 1 的转移概率如下所示,算例 2 和算例 3 相应的转移概率与算例 1 相似.

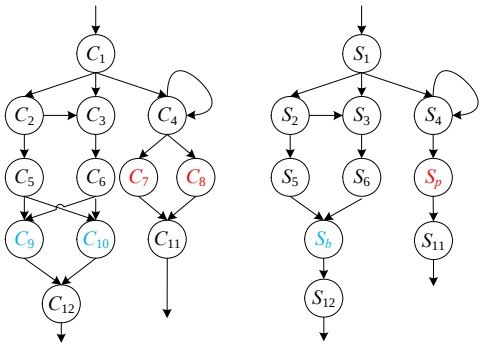
$$P_{1,2}=0.6 \quad P_{1,3}=0.1 \quad P_{1,4}=0.3 \quad P_{2,3}=0.3 \quad P_{2,5}=0.7 \quad P_{3,6}=1 \quad P_{4,4}=0.2$$
$$P_{4,p}=0.8 \quad P_{5,b}=1 \quad P_{6,b}=1 \quad P_{p,11}=1 \quad P_{b,12}=1 \quad P_{11,6}=0.1 \quad P_{11,12}=0.9$$



(a) 体系结构图 (b) 状态图

Fig.9.1 A single-input/single-output type (example 1)

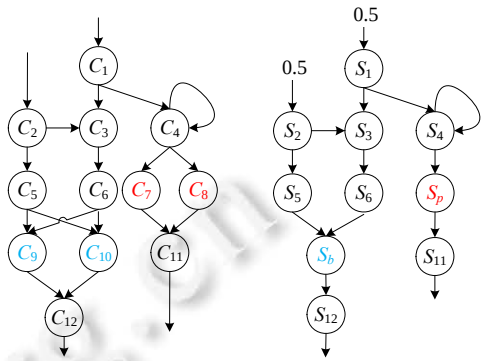
图 9.1 单输入单输出系统体系结构图和状态图(算例 1)



(a) 体系结构图 (b) 状态图

Fig.9.2 A single-input/multiple-output type (example 2)

图 9.2 单输入多输出系统体系结构图和状态图(算例 2)



(a) 体系结构图 (b) 状态图

Fig.9.3 A multiple-input/multiple-output type (example 3)

图 9.3 多输入多输出系统体系结构图和状态图(算例 3)

如下对算例 1 进行详细分析,首先依据公式(3)将可靠性优化模型转变为无约束求目标函数最小化:

$$\text{Min} \quad cost + penalty = a_i \cdot \sum_{i=1}^{n} e^{(1-c_i)\cdot \frac{r_i - r_{i,\min}}{r_{i,\max} - r_i}} + k \cdot \min\{0, RE - r_0\} \tag{30}$$

其中,a=[0.88 0.9 0.86 0.76 0.89 0.93 0.94 0.75 0.95],c=[0.23 0.28 0.37 0.45 0.05 0.25 0.5 0.2 0.41],$r_{i,\min}$=0.5, $r_{i,\max}$=1,r_0=0.9.

根据第 3 节中不同体系风格可靠性的分析,算例 1 软件系统可靠性表示为公式(31):

$$\left.\begin{aligned} RE = {} & 0.42 \cdot r(c_1) \cdot r(c_2) \cdot r(c_5) \cdot r(c_b) \cdot r(c_{12}) + 0.18 \cdot r(c_1) \cdot r(c_2) \cdot r(c_3) \cdot r(c_6) \cdot r(c_b) \cdot r(c_{12}) + \\ & 0.1 \cdot r(c_1) \cdot r(c_3) \cdot r(c_6) \cdot r(c_b) \cdot r(c_{12}) + \frac{0.216 \cdot r(c_1) \cdot r(c_4) \cdot r(c_p) \cdot r(c_{11}) \cdot r(c_{12})}{1 - 0.2 \cdot r(c_4)} + \\ & \frac{0.24 \cdot r(c_1) \cdot r(c_4) \cdot r(c_p) \cdot r(c_{11}) \cdot r(c_6) \cdot r(c_b) \cdot r(c_{12})}{1 - 0.2 \cdot r(c_4)} \end{aligned}\right\} \tag{31}$$

其中,各个构件可靠性 r 为可靠性分配的自变量,系统决策者根据最终求得的构件可靠性值分配资源,从而达到复杂软件整体可靠性最大化和代价最小化.

3.2.2 实验结果和分析

本文主要在收敛速度以及精度方面对 PHEDA-SCDE,DE,SCDE,EDA 以及 DHOACP 算法进行比较,每次目标函数独立运行 20 次.实验经过 200 次迭代,分别得到各种算法最优解的平均值、标准差、最优值、最差值(见表 3)作为目标函数精度的评价结果.同时,根据一次独立运行的 CPU 时间作为算法效率的评价结果,运行时间是一个关键的理论问题[25].从表 3 中可以看出:对于单输入单输出软件系统分配问题(算例 1),PHEDA-SCDE 在求解质量上明显优于其余算法,在求解效率上明显优于 DHOACP,对于单输入多输出系统(算例 2)和多输入多输出系统(算例 3)的软件可靠性分配问题,该算法无论在求解效率还是在求解精度上都具有良好的效果;其次,本文对传统的差分进化算法进行自适应交叉改进,改进结果表明,SCDE 算法在求解质量上明显优于 DE 算法,进一步验证了 SCDE 算法相较于 DE 算法全局搜索能力强,能迅速得到解空间.

Table 3 The solutions of objective functions

表 3 目标函数实验结果

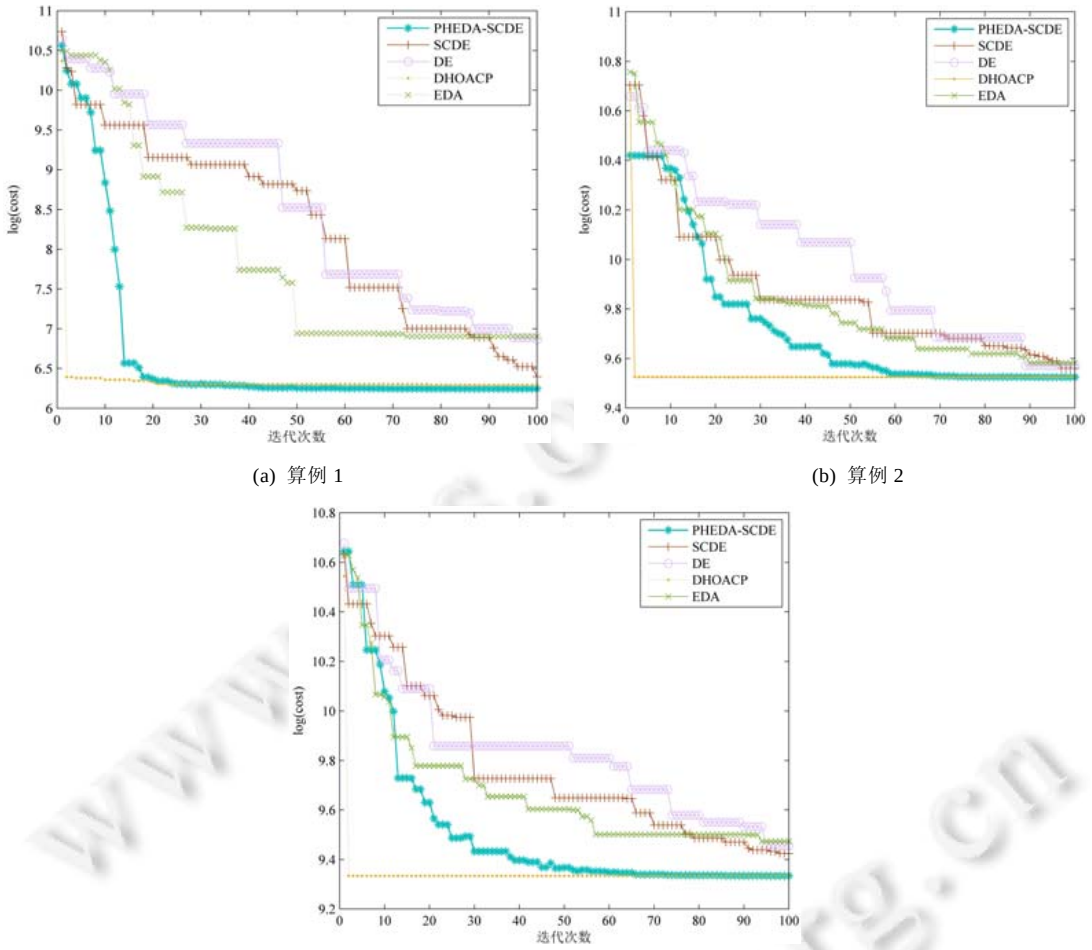
测试算例	算法	平均值	标准差	最优值	最差值	时间(s)
算例 1	PHEDA-SCDE	**515.652 9**	0.116 1	515.438 2	515.945 8	0.775 6
	DE	525.787 5	2.480 7	521.684 5	530.882 0	0.248 6
	SCDE	516.325 1	0.340 1	515.706 0	517.068 2	0.221 5
	EDA	1 320.637	1.85e+03	558.818 9	7 685.552	0.163 7
	DHOACP	533.042 5	5.585 4	524.793 9	542.719 2	19.602
算例 2	PHEDA-SCDE	**1.3692e+04**	4.95e−06	1.3692e+04	1.3692e+04	0.868 4
	DE	1.3694e+04	0.209 3	1.3693e+04	1.3694e+04	0.234 8
	SCDE	1.3693e+04	3.85e−04	1.3693e+04	1.3693e+04	0.224 0
	EDA	1.6623 e+04	3.10 e+03	1.3914 e+04	2.4159 e+04	0.305 2
	DHOACP	1.3694e+04	0.302 1	1.3693e+04	1.3694e+04	18.525
算例 3	PHEDA-SCDE	**1.1315e+04**	6.20e−06	1.1315e+04	1.1315e+04	0.717 0
	DE	1.1316e+04	0.293 1	1.1315e+04	1.1316e+04	0.187 9
	SCDE	**1.1315e+04**	1.76e−04	1.1315e+04	1.1315e+04	0.186 3
	EDA	1.3312 e+04	2.39e+03	1.1713 e+04	2.2824 e+04	0.176 0
	DHOACP	1.1316e+04	0.289 3	1.1315e+04	1.1316e+04	22.347

收敛曲线图主要描绘随着进化次数的增加目标函数适应度值的变化趋势,算法收敛曲线图如图 10 所示.图 10(a)为算例 1 的算法收敛曲线图,图 10(b)为算例 2 的算法收敛曲线图,图 10(c)为算例 3 的算法收敛曲线图.为了保证系统可靠性,软件代价不会是一个无穷小的值,一定收敛于某个值.为了直观地描述软件代价的收敛曲线,算法收敛曲线图纵坐标表示适应度值的对数变换,值为 $\log_{10}(f(x))$.横坐标表示进化代数.从 3 幅图中可以看出,DHOACP 算法虽然在其初期收敛速度最快,但很快陷入局部最优.在算法收敛性上,PHEDA-SCDE 明显优于其他 3 种算法,说明算法在初期快速搜索解空间,有很强的全局搜索能力.因此,PHEDA-SCDE 算法结合分布估计算法和自适应交叉差分进化算法的优点,无论在求解精度还是收敛速度方面,对于复杂软件系统可靠性的分配均具有优秀的效果.算法详细参数设置见表 4.

Table 4 Parameter tuning of all compared algorithms

表 4 算法运行参数设置

算法	参数类型	参数名称	值
PHEDA-SCDE	CR_0	交叉算子	0.2
	CR_1	交叉算子	0.8
	F_0	初始变异算子	0.6
	truncation	阈值	0.3
	$n_{\max}$	决策因子上限	0.2
	$n_{\min}$	决策因子下限	1.0
DE	CR	交叉算子	0.2
	F_0	初始变异算子	0.6
SCDE	CR_0	交叉算子	0.2
	CR_1	交叉算子	0.8
	F_0	初始变异算子	0.6
EDA	*truncation*	阈值	0.3
DHOACP	w	惯性权重	0.8
	C_1	加速常数	1.5
	C_2	加速常数	1.5
	$V_{\max}$	粒子最大速度	1
	$V_{\min}$	粒子最小速度	0



(a) 算例 1 (b) 算例 2

(c) 算例 3

Fig.10 The convergence curve of all compared algorithms

图 10 算法收敛曲线图

4 总 结

可靠性问题一直是软件可靠性领域的重中之重的研究难题,合理分配构件可靠性有利于增强系统可靠性、提高资源利用率、减少开发维护成本.对于大型复杂软件系统,本文提出一种新型的元启发式算法——基于罚函数的混合分布估计和自适应交叉差分进化的优化算法(PHEDA-SCDE),该方法结合分布估计算法和差分进化算法的优点.在算法初期,PHEDA-SCDE 具有全局搜索能力强、收敛速度快;在算法中后期,该算法具有较强的局部搜索能力,避免对复杂优化问题陷入局部最优.实验结果证明,算法以更少的成本、较短的计算时间优于其他现有算法.

未来的研究方向主要是云计算环境下的软件可靠性评估和分配问题,提高算法的精度和收敛速度,改进目标函数的准确性,研究构件可靠性对系统可靠性的影响.

References:

[1] Kuo W, Prasad VR. An annotated overview of system reliability optimization. IEEE Trans. on Reliability, 2000,49(2):176–187.

[2] Ravi V, Reddy PJ, Zimmermann HJ. Fuzzy global optimization of complex system reliability. IEEE Trans. on Fuzzy Systems, 2000, 8(3):241–248.

[3] Nakagawa Y, Nakashima K. A heuristic method for determining optimal reliability allocation. IEEE Trans. on Reliability, 2009, R-26(3):156−161.

[4] Ramirez-Marque JE, Rocco CM. Evolutionary optimization technique for multi-state two-terminal reliability allocation in multi-objective problems. IIE Transactions, 2010,42(8):539−552.

[5] Hsieh TJ, Yeh WC. Penalty guided bees search for redundancy allocation problems with a mix of components in series-parallel systems. Computers & Operations Research, 2012,39(11):2688−2704.

[6] Khalili-Damghani K, Abtahi AR, Tavana M. A new multi-objective particle swarm optimization method for solving reliability redundancy allocation problems. Reliability Engineering & System Safety, 2013,111(2):58−75.

[7] Tran DH, Cheng MY, Cao MT. Hybrid multiple objective artificial bee colony with differential evolution for the time−cost−quality trade off problem. Knowledge-Based Systems, 2015,74(1):176−186.

[8] Zhang J, Lei H. A pre-distribution algorithm of component reliability in Internetware system. Computing, 2015,97(7):755−768.

[9] Zhang E, Chen Q. Multi-Objective reliability redundancy allocation in an interval environment using particle swarm optimization. Reliability Engineering & System Safety, 2016,145:83−92.

[10] Goševa-Popstojanova K, Trivedi KS. Architecture-based approach to reliability assessment of software systems. Performance Evaluation, 2001,45(2-3):179−204.

[11] Wang WL, Pan D, Chen MH. Architecture-Based software reliability modeling. Journal of Systems & Software, 2006,79(1): 132−146.

[12] Cheung RC. A user-oriented software reliability model. IEEE Trans. on Software Engineering, 1980,6(2):118−125.

[13] Helander ME, Zhao M, Ohlsson N. Planning models for software reliability and cost. IEEE Trans. on Software Engineering, 1998, 24(6):420−434.

[14] Li F, Wu T, Hu M, Dong J. An accurate penalty-based approach for reliability-based design optimization. Research in Engineering Design, 2010,21(2):87−98.

[15] Mettas A. Reliability allocation and optimization for complex systems. In: Proc. of the Annual Reliability and Maintainability Symp. 2000,30(6):216−221.

[16] Wang SY, Wang L, Xu Y. An effective estimation of distribution algorithm for the flexible job-shop scheduling problem with fuzzy processing time. Int'l Journal of Production Research, 2013,51(12):3778−3793.

[17] Das S, Suganthan PN. Differential evolution: A survey of the state-of-the-art. IEEE Trans. on Evolutionary Computation, 2011, 15(1):4−31.

[18] Qin AK, Huang VL, Suganthan PN. Differential evolution algorithm with strategy adaptation for global numerical optimization. IEEE Trans. on Evolutionary Computation, 2009,13(2):398−417.

[19] Wang S, Li Y, Yang H, Liu H. Self-Adaptive differential evolution algorithm with improved mutation strategy. Applied Intelligence, 2017,6:1−15.

[20] Alguliev RM, Aliguliyev RM, Isazade NR. DESAMC+DocSum: Differential evolution with self-adaptive mutation and crossover parameters for multi-document summarization. Knowledge-Based Systems, 2012,36(36):21−38.

[21] Lipowski A, Lipowska D. Roulette-Wheel selection via stochastic acceptance. Physica A Statistical Mechanics & Its Applications, 2011,391(6):2193−2196.

[22] Zhang B, Pan QK, Gao L, Zhang XL, Sang HY, Li JQ. An effective modified migrating birds optimization for hybrid flowshop scheduling problem with lot streaming. Applied Soft Computing, 2016,52(C):14−27.

[23] Kempthorne O, Montgomery DC. Design and analysis of experiments. Journal of the American Statistical Association, 2008, 101(474):853−854.

[24] Lo JH, Huang CY, Chen IY, Kuo SY, Lyu MR. Reliability assessment and sensitivity analysis of software reliability growth modeling based on software module structure. Journal of Systems & Software, 2005,76(1):3−13.

[25] Yu Y, Qian C. Running time analysis: Convergence-based analysis reduces to switch analysis. In: Proc. of the IEEE Congress on Evolutionary Computation. New York :IEEE, 2015. 2603−2610.

附录 A

通过一个简单的算例来介绍 PHEDA-SCDE 算法的进化操作,对其有一个直观的认识.假设用分布估计算法求解 $f(x_1,x_2,x_3)=x_1+x_2+x_3$ 的最大值,其中,$x_j\in[0,1](j=1,2,3)$.

第 1 步,依据公式(5)初始化种群 $x,x_i=\{x_{i,1},x_{i,2},\ldots,x_{i,D}\},i=1,2,\ldots,NP$.种群大小 NP 为 5,种群维度 D 为 3,通过适应度函数 $f(x_1,x_2,x_3)=x_1+x_2+x_3$ 计算各个个体适应度值,此时,最优个体为个体 1.初始种群及适应度值见表 5.

Table 5 The initial population and its fitness

表 5 初始种群及适应度值

个体	x_1	x_2	x_3	f
1	0.2	0.8	0.9	1.9
2	0.4	0.1	0.2	0.7
3	0.6	1	0	1.6
4	0.3	0.1	0.5	0.9
5	0.7	0.1	0.7	1.5

第 2 步,通过 EDA 算法对初始种群进行优化.对种群按照适应度值从高到低排序,假设阈值 $truncation$=0.6,选择较优的 $truncation\times NP$=3 个个体,分别为个体 1、个体 3 和个体 5.

a) 根据均值表达式(32)和方差表达式(33),得到较优种群的均值和方差:

$$u_j=\frac{\sum_{i=1}^{NP}x_{ij}}{NP} \tag{32}$$

$$\sigma_j=\sqrt{\frac{\sum_{i=1}^{NP}(x_{ij}-u_j)^2}{NP}} \tag{33}$$

b) 根据公式(34)构建概率模型,并从中采样,得到 NP 个新样本,构成新种群:

$$P(x)=\prod_{j=1}^{D}\frac{1}{\sqrt{2\pi}\sigma_j}\mathrm{e}^{\frac{1}{2}\left(\frac{x_j-u_j}{\sigma_j}\right)^2} \tag{34}$$

算法 EDA 生成的实验种群及适应度值见表 6.

Table 6 The EDA population and its fitness

表 6 EDA 实验种群及适应度值

个体	u_1	u_2	u_3	f
1	0.2	0.8	0.9	1.9
3	0.6	1	0	1.6
5	0.7	0.1	0.7	1.5
6	0.5	0.7	0.4	1.6
7	0.6	0.1	0.8	1.5

c) 假设此时为算法初期,决策因子 n=4.每个个体被选中的概率与适应度值大小成正比.适应度值越大,被选中的概率越高.此时,采用轮盘赌方法选择 4 个适应度值较高的实验个体,组成 EDA 候选实验种群,其种群及适应度值见表 7.

Table 7 The EDA candidate population and its fitness

表 7 EDA 候选实验种群及适应度值

个体	u_1	u_2	u_3	f
1	0.2	0.8	0.9	1.9
3	0.6	1	0	1.6
5	0.7	0.1	0.7	1.5
7	0.6	0.1	0.8	1.5

第 3 步,通过 SCDE 算法对初始种群进行优化.

a) 假设初始变异算子 F_0=0.1,当前迭代次数为 1,依据公式(6)自适应变异生成变异个体 v,例如 $v_2=x_2+0.2\times$

(x_5–x_4).变异种群及适应度值见表 8.

Table 8 The SCDE mutation population and its fitness

表 8 SCDE 变异种群及适应度值

个体	v_1	v_2	v_3	f
1	0	1	1	2
2	1	0.1	0.6	1.7
3	1	1	0.4	2.4
4	0.1	1	1	2.1
5	0.5	1	1	2.5

b) 假设交叉算子 CR_0=0.4,CR_1=0.7,依据公式(9)和图 1 进行自适应交叉,生成实验个体 u.SCDE 实验种群及适应度值见表 9.

Table 9 The SCDE population and its fitness

表 9 SCDE 实验种群及适应度值

个体	u_1	u_2	u_3	f
1	0.2	0.8	0.9	1.9
2	1	0.1	0.6	1.7
3	0.2	0.8	0.9	1.9
4	0.5	0.7	0.4	1.6
5	0.5	1	1	2.5

c) 根据第 2 步中假设的决策因子 n=4,此时,轮盘赌选择 1 个适应度值较高的实验个体 5,组成 SCDE 候选实验种群.

第 4 步,合并 EDA 的候选实验个体 1,3,5,7 和 SCDE 的候选实验个体 5 为新种群,其种群及适应度值见表 10.

Table 10 The new population and its fitness

表 10 新种群及适应度值

个体	u_1	u_2	u_3	f
1	0.2	0.8	0.9	1.9
2	0.6	1	0	1.6
3	0.7	0.1	0.7	1.5
4	0.6	0.1	0.8	1.5
5	0.5	1	1	2.5

第 5 步,依据公式(8)对当前种群和新种群进行选择,选择更优向量产生下一代.

第 6 步,依据公式(10)更新决策因子 n.

第 7 步,若终止条件满足,则算法终止;否则,转向第 2 步继续执行.

徐悦(1994－),女,江苏苏州人,博士生,主要研究领域为软件可靠性,智能优化算法.

皮德常(1971－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为数据挖掘,系统优化.