文章编号: 1000- 0887(2008) 12- 1486- 09

大型网络基于连通可靠度的最优可靠度分配*

陈玲俐, 于 洁

(上海大学 土木系, 上海 200072)

(叶志明推荐)

摘要: 由于网络连通可靠度计算属于 NP- hard 问题, 当系统可靠度无法显式表达时, 基于连通可靠度的大型复杂网络优化通常只能采用启发式优化算法解决. 通过对复杂网络连通可靠度算法结构的分析, 给出了系统连通可靠度的 Taylor 方程. 采用遗传算法, 由系统连通可靠度的 Taylor 方程确定种群适应值, 得到一个系统最优可靠度分配方案; 将最优解带入改进 Minty 算法或递推分解算法中, 计算该最优解的连通可靠度精确值和对应的连通可靠度的 Taylor 展开方程; 再次采用遗传算法求最优解. 当最优解对应的可靠度精确值和 Taylor 方程算得得近似值误差小于指定精度时, 则此最优解为最终的系统最优可靠度分配方案. 将此优化过程称为迭代遗传算法. 算例显示迭代遗传算法不仅可用于大型网络的连通可靠度最优分配, 而且优化迭代过程中可以得到多组阶段最优解, 这些解均落在最优解附近, 构成了近似最优解群, 在实际工程优化中拓展了选择面.

关 键 词: 最优可靠度分配; 连通可靠度; 遗传算法; 改进 Minty 算法; 递推分解算法
中图分类号: O22; TP202.1 文献标识码: A

引 言

系统的可靠度是由组成系统的单元可靠度决定的. 简单的串联、并联系统的整体可靠度可以表达为单元可靠度的显式函数, 可以直接将单元的可靠度作为系统可靠度优化的优化参数, 用数学优化算法求解系统最优的可靠度分配方案. 对于复杂的网络系统, 系统的可靠度无法表达为单元可靠度的显式函数, 无法用传统数学优化方法对系统可靠度进行优化.

大型复杂网络系统的可靠度计算本身就是一个 NP- hard 问题; 而系统最优可靠度分配为组合优化问题, 也存在计算复杂性困难. 寻求高效的优化算法成为大型复杂网络系统可靠度优化的一个研究热点. 文献[1] 总结了近半个世纪以来的系统的可靠度优化方法, 将这些方法归纳为 5 类: 1) 超启发式优化方法, 包括蚁群算法(ACO)、遗传算法(GA)、禁忌搜索(TS)、模拟退火(SA)、免疫算法(IA)、暴雨算法(GDA)、细胞进化方法(CEA) 等; 2) 精确优化法, 如分枝定界法; 3) 最大- 最小迫近法; 4) 启发式优化方法; 5) 动态规划法. 各类系统可靠度优化方法进一步对比分析可以参见文献[2- 4] . 其中, 超启发式优化方法适用于复杂系统, 优化效率和

* 收稿日期: 2008- 01-04; 修订日期: 2008-09-02
基金项目: 上海市教育委员会科研资助项目(05AZ74); 上海市科学技术委员会科研资助项目(04JC14035)
作者简介: 陈玲俐(1972—), 女, 新疆石河子人, 副教授, 博士(联系人. Tel: + 86- 21- 56135149; E- mail: ChenLingli72@sina.com).

优化质量均较高,是目前世界上应用最广的一类可靠度优化方法.

为了更好地解决系统可靠度优化问题,国内外学者多是通过改进优化算法来提高系统可靠度优化效率. 在他们给出的算例中,系统可靠度已被表达为单元可靠度的显式方程. 当系统复杂到可靠度无法显式表达时,系统可靠度优化的困难主要来自在优化的每一步需要重新确定系统可靠度. 本文通过对复杂网络可靠度算法结构的分析,得到了系统可靠度对单元可靠度的一阶和二阶导数表达式;至此,原本无法显式表达的复杂系统可靠度,被成功表述为一阶和二阶Taylor展开式;Taylor系数推荐采用改进的Minty算法[5-6]或递推分解算法[7-8]计算. 基于系统可靠度的Taylor展开方程可以采用传统优化方法或超启发式优化方法进行系统可靠度优化. 由于Taylor展开方程是系统可靠度的近似描述,为了提高计算精度,本文采用了迭代遗传算法[9-10]实现了网络系统的最优可靠度分配.

1 复杂网络连通可靠度算法结构

这里选择单源单汇网络图进行其连通可靠度算法结构的分析,不会破坏其普遍性和基本原理. 我们将初始网络记为G0. G0中从源点到达汇点的所有路径可以分为包含单元 i 的路径和不包含单元 i 的路径,这两类路径显然是正交的. G0网络的系统可靠度可以记为:

$$R_{\mathrm{sys}} = r_i \cdot R_{\mathrm{sys}}(r_1, r_2, \cdots, r_{i-1}, 1, r_{i+1}, \cdots, r_n) + (1 - r_i) \cdot R_{\mathrm{sys}}(r_1, r_2, \cdots, r_{i-1}, 0, r_{i+1}, \cdots, r_n), \tag{1}$$

r_i 为单元 i 的可靠度.

如果将单元 i 的始末节点融合为同一个节点,可以得到一个新的网络,记为G1. G1中连通源点汇点的所有路径均包含单元 i. 因此,$R_{\mathrm{sys}}(r_1, r_2, \cdots, r_{i-1}, 1, r_{i+1}, \cdots, r_n)$ 就对应网络图G1的系统可靠度.

如果在网络G0中删除单元 i 后同样也可以得到一个新的网络图,记为G2. G2中的所有路径均不包含单元 i. 所以 $R_{\mathrm{sys}}(r_1, r_2, \cdots, r_{i-1}, 0, r_{i+1}, \cdots, r_n)$ 对应网络G2的系统可靠度. G1和G2可以视为G0的子网络,子网络比其父网络结构更加简单. G1和G2则构成正交的兄弟网络. 方程(1)此时可以另写为

$$R_{\mathrm{sys}} = r_i \cdot R_{\mathrm{sys}}^{\mathrm{G1}} + (1 - r_i) \cdot R_{\mathrm{sys}}^{\mathrm{G2}}, \tag{2}$$

$$\frac{\partial R_{\mathrm{sys}}}{\partial r_i} = R_{\mathrm{sys}}^{\mathrm{G1}} - R_{\mathrm{sys}}^{\mathrm{G2}}, \tag{3a}$$

$$\frac{\partial^2 R_{\mathrm{sys}}}{\partial r_i^2} = 0. \tag{3b}$$

一旦求得了子网络的系统可靠度,则其父网络的系统可靠度可以推算出来. 而G1、G2的系统可靠度可以用相同的递推方法得到:

$$R_{\mathrm{sys}}^{\mathrm{G1}} = r_j \cdot R_{\mathrm{sys}}^{\mathrm{G11}}(r_1, r_2, \cdots, r_{j-1}, 1, r_{j+1}, \cdots, r_n) + (1 - r_j) \cdot R_{\mathrm{sys}}^{\mathrm{G12}}(r_1, r_2, \cdots, r_{j-1}, 0, r_{j+1}, \cdots, r_n),$$

$$R_{\mathrm{sys}}^{\mathrm{G2}} = r_j \cdot R_{\mathrm{sys}}^{\mathrm{G21}}(r_1, r_2, \cdots, r_{j-1}, 1, r_{j+1}, \cdots, r_n) + (1 - r_j) \cdot R_{\mathrm{sys}}^{\mathrm{G22}}(r_1, r_2, \cdots, r_{j-1}, 0, r_{j+1}, \cdots, r_n);$$

$$\frac{\partial R_{\mathrm{sys}}}{\partial r_j} = r_i \frac{\partial R_{\mathrm{sys}}^{\mathrm{G1}}}{\partial r_j} + (1 - r_i) \frac{\partial R_{\mathrm{sys}}^{\mathrm{G2}}}{\partial r_j} = r_i (R_{\mathrm{sys}}^{\mathrm{G11}} - R_{\mathrm{sys}}^{\mathrm{G12}}) + (1 - r_i)(R_{\mathrm{sys}}^{\mathrm{G21}} - R_{\mathrm{sys}}^{\mathrm{G21}}), \tag{4a}$$

$$\frac{\partial^2 R_{\mathrm{sys}}}{\partial r_i \partial r_j} = \frac{\partial R_{\mathrm{sys}}^{\mathrm{G1}}}{\partial r_j} - \frac{\partial R_{\mathrm{sys}}^{\mathrm{G2}}}{\partial r_j} = (R_{\mathrm{sys}}^{\mathrm{G11}} - R_{\mathrm{sys}}^{\mathrm{G12}}) - (R_{\mathrm{sys}}^{\mathrm{G21}} - R_{\mathrm{sys}}^{\mathrm{G21}}); \tag{4b}$$

r_j 为单元 j 的可靠度. 单元 j 为新的分解边.

在新的网络图中不断分解下去,得到的子网络的拓扑结构越来越简单. 当所有的边都分解完毕时,将子网络的系统可靠度回代到式(2)中,就得到了父网络的系统可靠度;逐级回代后,由式(2)~(4)我们不仅可以得到初始网络 G0 的系统连通可靠度,而且可以得到系统可靠度相对单元可靠度的一阶和二阶导数. 系统连通可靠度计算结构可以用一个二分图来表示(图 1).

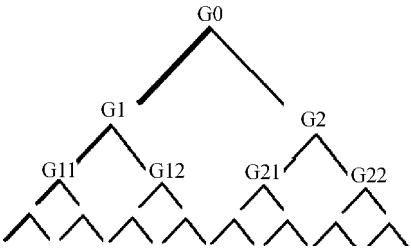

图 1 网络连通可靠度计算结构图

如果我们机械地按照此结构分解计算下去,则每分解一步子图数量就增加一倍,计算和存储都将无法进行下去. 这里首先介绍改进 Minty 算法所采用的分解计算方法[5-6],该方法曾被国外学者认为是最好的一种计算方法,也是目前应用最广的算法.

在改进的 Minty 算法中 G0 是沿一条枝干($G0^{\rightarrow}G1^{\rightarrow}G11^{\rightarrow}$)进行分解的,在分解过程中得到的子图 G2、G12 等放入堆栈中暂不分解,直到这条枝干分解完毕或者找到了一条路径. 如果我们能够判断一个网络不存在连通源点和汇点的路径,则该网络不必继续分解;如果分解边已经构成一条连接源点和汇点的路径,则可以中止分解;且这条路径的连通可靠度等于所有分解边概率的连乘. 所有连通路径组成集合记为 S_{road},则

$$R_{\text{road}} = \prod P(e_i), \qquad e_i \in S_{\text{road}}, \tag{5}$$

$$P(e_i) = \begin{cases} r_i, & e_i = 1, \\ 1 - r_i, & e_i = 0, \end{cases}$$

$$\frac{\partial R_{\text{road}}}{\partial r_i} = \frac{\partial \prod P(e_i)}{\partial r_i}, \qquad e_i \in S_{\text{road}}, \tag{6}$$

$$\frac{\partial R_{\text{road}}}{\partial r_i \partial r_j} = \begin{cases} \dfrac{\partial \prod P(e_i)}{\partial r_i \partial r_j}, & e_i, e_j \in S_{\text{road}}, \\ 0, & e_i \text{或} e_j \notin S_{\text{road}}. \end{cases} \tag{7}$$

从堆栈中取出最后放入的子图进行分解,即在二分图中选择距离叶端最近的新的枝条进行分解. 这种分解计算方法使计算过程中子图的存储数量永远不会超过系统可分解单元的数量[6].

在图 1 中找到的所有路径正交,因此

$$R_{\text{sys}} = \sum R_{\text{road}}, \tag{8}$$

$$\frac{\partial R_{\text{sys}}}{\partial r_i} = \sum \frac{\partial R_{\text{road}}}{\partial r_i}, \tag{9}$$

$$\frac{\partial R_{\text{road}}}{\partial r_i \partial r_j} = \sum \frac{\partial R_{\text{road}}}{\partial r_i \partial r_j}. \tag{10}$$

系统可靠度与单元可靠度的关系可以近似表达为

$$R_{\text{sys}} \approx R_{\text{sys}}|_{r=r} + \boldsymbol{A}\begin{Bmatrix} r_1 - r_1 \\ \vdots \\ r_n - r_n \end{Bmatrix} + [r_1 - r_1, \cdots, r_n - r_n]\boldsymbol{B}\begin{Bmatrix} r_1 - r_1 \\ \vdots \\ r_n - r_n \end{Bmatrix}, \tag{11}$$

$$\boldsymbol{A} = \left[\frac{\partial R_{\text{sys}}}{\partial r_1}, \cdots, \frac{\partial R_{\text{sys}}}{\partial r_n}\right],$$

$$\boldsymbol{B}=\begin{bmatrix} 0 & \dfrac{\partial^2 R_{sys}}{\partial r_1\partial r_2} & \cdots & \dfrac{\partial^2 R_{sys}}{\partial r_1\partial r_n} \\ \dfrac{\partial^2 R_{sys}}{\partial r_2\partial r_1} & 0 & \cdots & \dfrac{\partial^2 R_{sys}}{\partial r_2\partial r_n} \\ \vdots & \vdots & \ddots & \vdots \\ \dfrac{\partial^2 R_{sys}}{\partial r_n\partial r_1} & \dfrac{\partial^2 R_{sys}}{\partial r_n\partial r_2} & \cdots & 0 \end{bmatrix};$$

或者
$$R_{sys}\approx R_{sys}\mid_{r=r}+\boldsymbol{A}\begin{Bmatrix} r_1-r_1 \\ \vdots \\ r_n-r_n \end{Bmatrix}. \tag{12}$$

采用正交分解思想来计算系统连通可靠度的算法有很多,但是所有这些方法的算法结构都遵循图 1 的计算结构;式(8)~(12)对这类方法均成立.这些算法的差别仅在于分解时分解枝干的选择有所不同,导致分解过程不同.比较后我们认为递推分解算法是该类算法中最好的计算方法[7-8],在超大型网络可靠度分析中其优势尤为突出.

递推分解算法总是选择图 1 中最短的枝条作为其分解枝干.枝条长度等于连通路径的长度,最短枝条对应网络的最短连通路径.因此在分解前需要采用宽度优先搜索(WFS)找出最短路径.递推分解算法计算效率很高,因为该算法所产生的子网络数量远小于其他算法.

递推分解算法的另一个优点在于超大型网络可靠度计算时可以实现指定精度(ε_1)下的近似计算.图 1 中得到的一条路径越长,则该路径对初始网络的系统可靠度的贡献相对较小,这在式(5)和式(8)中可以清楚地看到.在递推分解算法中优先处理网络的最短路和最小割集.这里的割集也是由分解边构成,因此递推分解算法所找到的所有割集也正交.将网络割集记为 S_{cut},初始系统的失效概率等于

$$P_{f,sys}=\sum P_{cut}, \tag{13}$$

$$P_{cut}=\prod P(e_i),\qquad e_i\in S_{cut}, \tag{14}$$

如果
$$|1-R_{sys}-P_{f,sys}|\leqslant\varepsilon_1,\ R_{sys}\approx R_{sys}. \tag{15}$$

系统可靠度和失效概率的计算是这些正交路径和正交割集概率的累加过程.R_{sys} 和 $P_{f,sys}$ 是分解过程中系统可靠度和失效概率的阶段结果.方程(15)意味着阶段结果已经满足计算精度,我们无需完成整个分解过程,这一点在大型网络可靠度计算和系统优化中非常有用.

2 系统最优可靠度分配模型

传统的系统最优可靠度分配模型为

$$\text{P1: } \max R_{sys}, \tag{16}$$

$$\text{sub}\begin{cases} \sum c_i(r_i)\leqslant C_0, \\ r_{min}\leqslant r_i<1. \end{cases} \tag{17}$$

R_{sys} 为系统的连通可靠度,是系统单元可靠度的函数;c_i 为单元 i 的造价,是单元可靠度 r_i 的函数;r_{min} 为单元允许的最小可靠度.

由于系统可靠度计算费时,我们不能在优化的每一步重新计算系统可靠度.因此我们将优化模型改写为

$$\text{P2:}\begin{cases}\text{P3:}\begin{cases}\max R_{\text{sys}},\\ \text{sub}\begin{cases}\sum c_i(r_0+\Delta r)\leqslant C_0,\\ r_{\min}\leqslant r_0+\Delta r<1,\end{cases}\end{cases}\\ R_{\text{sys}}(r^*)-R_{\text{sys}}(r_0)\leqslant\varepsilon, & \text{停止},\\ \text{否则} & r_0=r^*.\end{cases}\tag{18}$$

R_{sys} 为采用改进 Minty 算法或递推分解算法得到的系统可靠度, R_{sys} 为式(11)或(12)得到的系统可靠度近似表达方程. r^* 为优化模型 P3 的优化结果. 如果 R_{sys} 采用一阶 Taylor 展开式(12)计算, r^* 可以采用传统线性优化算法得到; 如果 R_{sys} 采用二阶 Taylor 展开式(11)计算, 因为式(11)中的矩阵 $\boldsymbol{B}$ 不是正定矩阵, 优化模型不能采用传统二次规划算法计算. 本文采用遗传算法(GA)求解模型 P3. GA 适合解决组合优化问题, 本文的 GA 作了几点改进. 在遗传操作中采用了新的选择机制, 并加入了记忆功能; 在杂交操作中采用了模糊双编码方法[11]; 优化中采用了动态种群规模. 我们将改进的遗传算法命名为有记忆双重模糊编码遗传算法, 算法细节在优化步骤中介绍.

3 优化步骤

优化步骤如下:

步骤 1 采用改进的 Minty 算法, 带入初始单元可靠度计算系统连通可靠度 $R_{\text{sys}}(r^{(0)})$ 及系数矩阵 $\boldsymbol{A}$、$\boldsymbol{B}$, 计算初始系统的整体造价. 初始单元可靠度构成第一条染色体.

步骤 2 设优化单元数目为 N, 初始种群规模为 P_0, $P_{\min}\leqslant P_0<P_{\max}$, $P_{\min}$、$P_{\max}$ 为指定的最小、最大种群规模. 由随机数发生器得到 $N\times P_0$ 个随机数. 一个单元的可靠度对应一个基因.

$$r_{ij}=r_{j,\min}+\text{random}(\cdot)\cdot(0.999\,99-r_{j,\min}),\tag{19}$$

$$\Delta r_{ij}=r_{ij}-r_j^{(0)},\tag{20}$$

r_{ij} 为第 i 条染色体的第 j 个基因. 每 N 个单元可靠度构成一条染色体, 共有 P_0 条染色体. 在这里采用的是实数编码.

步骤 3 采用 Taylor 展开式计算初始种群的系统可靠度近似值 R_{sys}^i 及系统造价 C_{sys}^i, $i=2\sim P_0$.

$$C_{\text{sys}}^i=\sum_{j=1}^{N}c_{ij},\tag{21}$$

$$c_{ij}=f(r_{ij}).\tag{22}$$

步骤 4 构建种群的适应值评价模型, 计算每条染色体的适应值.

第 i 条染色体的适应值为

$$f_{\text{fit}}(i)=\begin{cases}R_{\text{sys}}^i-\boldsymbol{A}\left\{\dfrac{c_{ij}}{\mathrm{d}c_{ij}/\mathrm{d}r_{ij}}\right\}\Big|_{j=1,\ldots,n}\dfrac{C_{\text{sys}}^i-C_0}{C_{\text{sys}}^i}, & C_{\text{sys}}^i>C_0,\\ R_{\text{sys}}^i, & C_{\text{sys}}^i\leqslant C_0.\end{cases}\tag{23}$$

步骤 5 选择操作.

在遗传操作中有多种选择方法在种群中选择可遗传染色体, 如赌盘选择、确定性原则、有退还或无退还随机选择等[9]. 在这些选择机制中适应值越大的染色体在下一代中留下的种子越多; 适应值相对较小的染色体则会在下一代灭绝. 本文采用的选择机制是相对适应值大于

零的染色体会在下一代中留下一个种子．我们将选择操作后保留的种群称为精英群落．

种群的相对适应值定义为

$$f'_{\mathrm{fit}}(i) = \mathrm{ind}\left\{\frac{f_{\mathrm{fit}}(i) - \min\{f_{\mathrm{fit}}(j), j = 1, P_0\}}{\sigma\{f_{\mathrm{fit}}(j), j = 1, P_0\}}\right\}, \tag{24}$$

$$\sigma\{f_{\mathrm{fit}}(\cdot)\} = \frac{\sum |f_{\mathrm{fit}}(\cdot) - E|}{P_0}, \tag{25}$$

E 为当前种群适应值的均值．$\mathrm{ind}(\cdot)$ 为取整函数．σ 定义为种群适应值的评价差异值．

$f'_{\mathrm{fit}}(i) > 0$ 的染色体构成了子代种群．假定此时子代种群的数量为 n_1, n_1 满足

$$P_{\min} < n_1 < P_0 < P_{\max}. \tag{26}$$

假如 $n_1 < P_{\min}$, 则在已淘汰种群中采用式(24) 计算淘汰种群的相对适应值, 进行二次选择．如果 $n_1 > P_{\max}$, 则在选出的子代种群中采用式(24) 计算该种群的相对适应值, 进行二次选择, 直到方程(26) 成立．

本文采用相对适应值进行选择避免了计算费时的适应值排队运算．存活染色体的适应值将保存在记忆中, 避免重复计算．

步骤 6　交叉操作．

当采用实数编码时隐含在交叉操作中的并行运算机制几乎丧失殆尽．因为采用实数编码后染色体的长度等于优化参数个数, 远小于采用二进制编码的染色体长度．这里采用模糊二进制编码来部分恢复交叉操作中隐含的并行机制．

设实数基因 x_j 的取值区间为(θ_1, θ_2), 一个基因的模糊二进制编码$(I_1I_2I_3)$ 与实数编码的对应关系如图 2 所示．

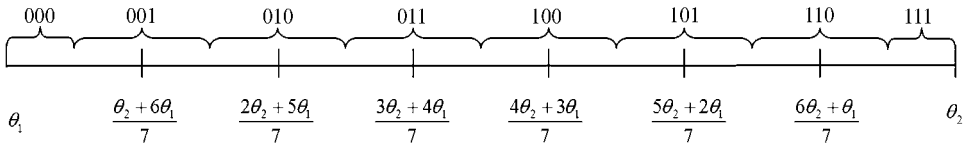

图 2　模糊二进制编码与实数区间的对应关系

采用模糊二进制编码进行交叉操作不仅将子代取值区间扩展到整个值域, 且交叉操作的并行计算效率为实数编码的 3 倍．虽然译码过程使交叉结果具有小幅的变异性, 但是父代解的大特征得以很好的传递和保留．

交叉操作次数为 $\mathrm{ind}(r_c \cdot n_1)$, r_c 为交叉率．这里为提高交叉操作的质量, 进行交叉的父代从选择出来的精英群落中随机挑选, 我们同时保留参与交叉的父代样本和子代染色体．交叉操作将产生 $2 \times \mathrm{ind}(r_c \cdot n_1)$ 个新染色体, 此时子代种群规模为 $n_1 + 2 \times \mathrm{ind}(r_c \cdot n_1)$．

步骤 7　变异操作．

在精英群落中随机选择染色体进行变异, 变异操作基于模糊二进制编码, 采用单点变异．变异后产生新的染色体, 旧的和新的染色体同时保留．

变异操作的次数为 $\mathrm{ind}(r_m \cdot n_1)$, r_m 为变异率．变异操作将产生 $\mathrm{ind}(r_m \cdot n_1)$ 个新解, 此时子代种群规模为 $P_1 = n_1 + 2 \times \mathrm{ind}(r_c \cdot n_1) + \mathrm{ind}(r_m \cdot n_1)$．

步骤 8　仅对子代中通过杂交、变异操作得到的新染色体进行译码, 并计算其近似可靠度、造价及对应的适应值．

根据图 2, 可以建立模糊二进制编码 $(I_1I_2I_3)$ 到实数编码的映射规则:

$$x_j=\begin{cases}\theta_1+\ \mathrm{rand}(\bullet)\ \dfrac{\theta_2-\ \theta_1}{14},\ I_1I_2I_3=\ 000,\\ \dfrac{6.5\theta_2+\ 0.5\theta_1}{7}+\ [\mathrm{rand}(\bullet)]\ \dfrac{\theta_2-\ \theta_1}{14},\ I_1I_2I_3=\ 111,\\ \text{否则},\ \theta_1+\ (4I_1+\ 2I_2+\ I_3)\ \dfrac{\theta_2-\ \theta_1}{7}+\ [\mathrm{rand}(\bullet)-\ 0.5]\ \dfrac{\theta_2-\ \theta_1}{7}.\end{cases}\tag{27}$$

步骤 9　判断模型 P3 是否收敛.

用公式(24)计算隔代精英种群的平均适应值的差异, 进行收敛判断:

$$N_{\mathrm{IT}}>\ N_{\mathrm{IT-max}},\ \text{或者},\ \sigma\leqslant\varepsilon_3,\tag{28}$$

$N_{\mathrm{IT-max}}$ 为最大遗传代数, ε_3 为一指定小正数.

如果式(28)不满足, 设 $P_0=\ P_1$, 转到步骤 5 继续遗传优化过程. 如果收敛准则满足, 转到步骤 10.

GA 用于连续变量优化问题时存在局部搜索能力差及欺骗风险. 为了提高 GA 的局部搜索效率、避免 GA 出现欺骗问题, 在文献[11]中我们提出了一个聚焦搜索策略. 由于 P3 模型仅为 P1 模型的近似模型, 模型 P3 的最优解还需要通过模型 P2 的检验才能成为最终的优化结果, 这期间需要多次采用 GA 反复优化. 这里得到的的最优解是否为 P3 模型的全局最优解会影响优化进程, 但是不影响最终优化结果. 因此, 这里无需采用聚焦搜索策略来提高 GA 的优化结果的质量.

步骤 10　判断模型 P2 是否收敛.

将 P3 的优化结果带入改进 Minty 算法或递推分解算法中, 计算系统可靠度 $R_{\mathrm{sys}}(r_{\mathrm{best}})$, 根据 Taylor 展开式中的系数矩阵 $\boldsymbol{A}$ 和 $\boldsymbol{B}$; 计算系统整体造价. 判断:

$$N_{\mathrm{NT}}>\ N_{\mathrm{NT-max}}\ \text{或者}\ R_{\mathrm{sys}}(r_{\mathrm{best}})-\ R_{\mathrm{sys}}(r^{(0)})<\ \varepsilon_2,\tag{29}$$

$N_{\mathrm{NT-max}}$ 为最大迭代次数, ε_2 为一指定小正数, $R_{\mathrm{sys}}(r_{\mathrm{best}})-\ R_{\mathrm{sys}}(r^{(0)})<\ \varepsilon_2$ 意味着 P3 的最优解与单元可靠度初始解相同.

如果式(29)不满足, 设

$$P_0=\ (P_{\min}+\ P_{\max})/2,\ r^{(0)}=\ r_{\mathrm{best}}.$$

转到步骤 2 继续迭代. 否则停止计算.

4　算　例

图 3 为一复杂网络, 该系统可靠度已经无法显式表达. 其中 1 点为源点, 6 点为汇点. 设

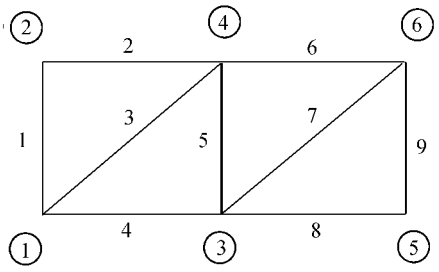

图 3　一网络拓扑结构

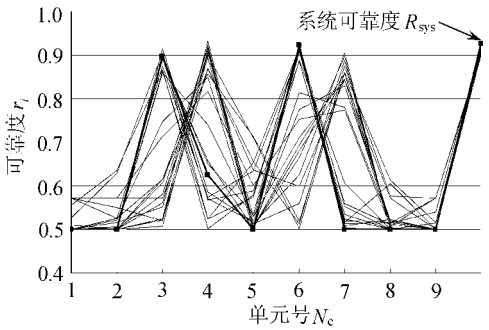

图 4　迭代遗传算法最后收敛种群
($C_0=\ 28, r_{\min}=\ 0.5$)

单元可靠度通过元件冗余方式得到提高. 单元可靠度与单元造价之间的对应关系为: $r = 1 - 0.7^c$. 设单元 $r_{\min} = 0.5$, 系统总造价限制为 $C_0 = 28$.

假定可靠度初值均为 0.5, 计算得到网络的初始系统可靠度 $R_{sys}^0 = 0.593\ 75$, Taylor 展开式中的矩阵:

$$\boldsymbol{A} = \{0.093\ 8,\ 0.093\ 8,\ 0.281\ 25,\ 0.367\ 187\ 5,\ 0.132\ 812\ 5,\ 0.367\ 187\ 5,\ 0.281\ 25,\ 0.093\ 8,\ 0.093\ 8\},$$

$$\boldsymbol{B} = \begin{bmatrix} 0 & 0.187\ 5 & -0.187\ 5 & -0.140\ 62 & 1.56\text{E}-2 & 0.109\ 38 & 0.00\text{E}+0 & 0.00\text{E}+0 & 0.00\text{E}+0 \\ & 0 & -0.187\ 5 & -0.140\ 62 & 1.56\text{E}-2 & 0.109\ 38 & 0.00\text{E}+0 & 0.00\text{E}+0 & 0.00\text{E}+0 \\ & & 0 & -0.421\ 88 & 4.69\text{E}-2 & 0.328\ 12 & 0.00\text{E}+0 & 0.00\text{E}+0 & 0.00\text{E}+0 \\ & & & 0 & -0.125 & -0.125 & 0.328\ 12 & 0.109\ 38 & 0.109\ 38 \\ & & & & 0 & -0.125 & 4.69\text{E}-2 & 1.56\text{E}-2 & 1.56\text{E}-2 \\ & & & & & 0 & -0.421\ 88 & -0.140\ 62 & -0.140\ 62 \\ & & 对称 & & & & 0 & -0.187\ 5 & -0.187\ 5 \\ & & & & & & & 0 & 0.187\ 5 \\ & & & & & & & & 0 \end{bmatrix}.$$

采用迭代遗传算法进行优化, 每次迭代得到的模型 P3 的优化结果见图 4. 图 4 中带有“■”标记的一组数值为单元近似灵敏度法得到的最优分配方案. 从图 4 中我们可以清楚看到每次迭代得到的模型 P2 的最优解对应的系统可靠度均在最优解附近, 可看作近似最优解.

本网络有两条最短路径(①→③→⑥, ①→④→⑥), 由于这两条路径是等价路径, 因此理论上该网络存在两个最优分配方案, 我们采用单元近似灵敏度法只能得到一个, 见表 1. 根据等价原则, 另一个最优解向量为{0.5, 0.5, 0.625 359, 0.897 59, 0.5, 0.5, 0.923 268, 0.5, 0.5}, 系统可靠度为 0.926 537.

表 1　　优化结果 ($C_0 = 28$, $r_{\min} = 0.5$)

参数	遗传算法		灵敏度法
整体造价	28.072 3	28.162 2	28*
系统可靠度	0.925 972	0.924 838	0.926 537*
r_1	0.5	0.5	0.5*
r_2	0.5	0.5	0.5*
r_3	0.915 212	0.605 496	0.897 59*
r_4	0.586 46	0.915 675	0.625 359*
r_5	0.517 311	0.529 381	0.5*
r_6	0.909 012	0.710 765	0.923 268*
r_7	0.534 236	0.845 568	0.5*
r_8	0.5	0.5	0.5*
r_9	0.5	0.5	0.5*

在图 4 中可以清楚看到近似最优解向两个理论最优解收敛. 采用迭代遗传算法得到的两组最优解和单元近似灵敏度法(*)得到的一组最优解见表 1. 虽然从表 1 中的计算结果看迭代遗传算法的优化结果比单元近似灵敏度法得到的优化结果略差. 但是迭代遗传算法在优化过程中可以得到多组近似最优的优化结果, 这意味着在实际工程中我们有多个选择方案.

[参　考　文　献]

[1] KOU Way, WAN Rui. Recent advances in optimal reliability allocation[J]. *IEEE Transactions on Systems, Man, and Cybernetics – Part A: Systems and Humans*, 2007, **37**(2): 143–156.

[2] Ravi Vadlamani. Modified great deluge algorithm versus other metaheuristics in reliability optimization[A].

In: *Studies in Computational Intelligence, Intelligence in Reliability Engineering (SCI)* [C]. Vol **40**. Berlin, Heidelberg: Springer, 2007, 21- 36.

[3] Lee H, Kuo W, Ha C. Comparison of max- min approach and NN method for reliability optimization of series - parallel system[J]. *Journal of System Science and Systems Engineering*, 2003, **12**(1): 39- 48.

[4] Ryoo H S. Robust meta- heuristic algorithm for redundancy optimization in large- scale complex systems [J]. *Annals of Operations Research*, 2005, **133**(1/4): 209- 228.

[5] Minty D J. A simple algorithm for listing all the trees of a graph[J]. *IEEE Trans, Finding Minimum Spanning Tree SIAM J Comput*, 1976, **5**(4): 724- 742.

[6] 陈树柏. 网络图论及其应用[M]. 北京: 科学出版社, 1982.

[7] 何军, 李杰. 大型生命线工程可靠度分析的递推分解算法[J]. 同济大学学报, 2001, **29**(7): 757- 762.

[8] Li J, He J. A recursive decomposition algorithm for the network seismic reliability evaluation[J]. *Earthquake Engineering and Structural Dynamics*, 2002, **31**(8): 1525- 1539.

[9] L•Z• 米凯利维兹. 演化程序——遗传算法和数据编码的结合[M]. 周家驹 译. 北京: 科学出版社, 2000.

[10] 陈玲俐. 城市供水系统抗震功能分析与优化[D]. 博士学位论文. 上海: 同济大学, 2002.

[11] 陈玲俐, 于洁. 双重模糊编码遗传算法及聚焦搜索策略[J/OL]. 中国科技论文在线, 2008. http://www.paper.edu.cn/downloadpaper.php?serial_number=200808-211&type=1.

Optimal Distribution of Reliability for Large Network Based on Connectivity

CHEN Ling- li, YU Jie

(Department of Civil Engineering, Shanghai University, Shanghai 200072, P. R. China)

Abstract: It is a non- polynomial complexity problem to calculate the connectivity of the complex network. When the reliability of the system can not be expressed as the function of the element reliability, some heuristic methods were applied to do the optimization based on connectivity of the network. The calculation structure of connectivity of complex network was analysed. The coefficient matrixes of Taylor second order expansion of the system connectivity was generated based on the calculation structure of connectivity of complex network. A optimal schedule is achieved based on genetic algorithms (GA). The fitness of seeds was calculated using the Taylor expansion function of system connectivity. The precise connectivity of the optimal schedule and the Taylor expansion function of system connectivity can be achieved by the approved Minty method or the recursive decomposition algorithm. When the error between the approximate connectivity and the precise value exceeds the assigned precision, optimize process was continued using GA and the Taylor function of system connectivity need renewal. The optimum process was called iterative GA. One case is illustrated iterative GA which can be used in the large network for optimal reliability attribution. One temporary optimal result will be generated every time in iteration process. These temporary optimal results approach the real optimal results. They can be regarded as the group of the approximate optimal results that is useful in the real project.

Key words: optimal distribution of reliability; connectivity; genetic algorithms(GA); approved Minty method; recursive decomposition algorithm