

遗传算法理论与实现技术

(二)

遗传算法的设计步骤

- 确定问题的编码方案。由于GA通常不直接作用于问题的解空间，而是利用解的某种编码表示来进行进化，因此选择合理的编码机制对算法质量和效率有很大影响。
- 确定适配值函数。由于GA通常基于适配值进行遗传操作，合理的适配值能够将各个体的优劣程度得以体现，并适应算法的进化过程。
- 遗传算子的设计。通常包括初始化、选择、交叉、变异和替换操作等。
- 算法参数的选取。通常包括种群数目、交叉概率、变异概率、进化代数等。
- 确定算法的终止条件。终止准则应根据所求解问题的性质，在优化质量和效率方面作合理均衡或侧重。

编码

- 编码就是将问题的解用一种码来表示，从而将问题的状态空间与GA的码空间相对应，这很大程度上依赖于问题的性质，并将影响遗传操作的设计。
- 由于GA的优化过程不是直接作用于问题参数本身，而是在一定编码机制对应的码空间上进行的，因此编码的选择是影响算法性能与效率的重要因素。

函数优化问题的编码

- 二进制编码、十进制编码、实数编码等
 - 二进制编码将问题的解用一个二进制0-1字符串来表示。
 - 譬如采用长度 l 为的二进制串 S 来表示 $[a, b]$ 区间内的变量 x ，则：
$$x = a + \frac{[S]_2}{2^l - 1} \cdot (b - a)$$
 - 十进制编码将问题的解用一个十进制串来表示。
 - 显然，不同的码长和码制，对问题求解的精度与效率有很大影响，而且算法也将付出较大的存储量和相应的转换运算。类似的还有 Gray 编码和串长可变的动态编码。
 - 实数编码将问题的解用一个实数来表示。
 - 它解决了二进制和十进制编码对算法精度和存储量的影响，同时便于优化中引入问题的相关信息，譬如梯度信息。实数编码直接在解的表现型上进行遗传操作，目前在高维复杂优化问题中得到广泛应用，并取得了较好效果。

组合优化问题的编码

- 鉴于组合优化问题本身的性质，其编码方式需要特殊设计。
- 常用的组合优化编码技术包括：
 - 置换排列
 - 0-1矩阵编码
 - ...
 - 后文将针对不同类型的调度问题进行介绍

适配值函数

- 适配值函数用于对个体进行评价，也是优化过程发展的依据。
- 对于简单的最小化问题，通常可以直接将目标函数变换成适配值函数(类似于后文的尺度窗口功能)，譬如将个体 X 的适配值 $f(X)$ 定义为 $M - c(X)$ 或 $e^{-ac(X)}$ ，其中 M 为一足够大正数， $c(X)$ 为个体的目标值， $a > 0$ 。
- 对于复杂优化问题，往往需要构造合适的评价函数，使其适应GA进行优化。

适配值函数(续)

- 由于适配值度量意义下的个体差异与目标函数值度量意义下的个体差异有所不同，因此若适配值函数设计不当，将难以体现个体的差异，选择操作的作用就很难体现出来，从而造成早熟收敛等缺点。
- 对适配值进行调节是常用的改进方法，如线性变换和指数变换等，即通过某种变换改变原适配值间的比例关系。
- 此外，目前已开发了许多基于序的选择操作，它基于目标值的好坏分配选择概率，无需设计适配值函数，一定程度上缓解了适配值函数定义的难度。
- 值得一提的是，GA的优化过程是一种搜索-评价过程，如果评价过程(即适配值或目标值的计算)占有时间过多，则势必影响算法的整体优化性能，因此对于评价过程复杂的问题，如仿真优化问题，如何提出有效的评价机制来加速或简化评价过程，将有利于提高GA的优化效率。

算法参数

- Grefenstette(1986)对求解函数优化的标准GA定义了6个关键参数, 包括:
 - 种群数目(population size, 记作N)
 - 交叉概率(crossover rate, 记作C)
 - 变异概率(mutation rate, 记作M)
 - 代沟(generation gap, 记作G)
 - 尺度窗口(scaling window, 记作W)
 - 选择策略(selection strategy, 记作S)
- 因此, 当GA框架确定后, 一种GA实现即对应一种(N、C、M、G、W、S)参数组合。
- 目前, 如何确定GA的最优参数仍旧是一个open问题, 也是研究GA的重要课题之一。
- 除了基于大量实验或经验结果确定合适参数外(即所谓的参数调整, parameter tuning), 许多学者设计自适应参数来动态调整GA的搜索行为和能力(即所谓的参数控制, parameter control), 而理论性的结论至今仍旧很少。

种群数目N

- 种群数目是影响算法最终优化性能和效率的因素之一。
- 通常，种群太小则不能提供足够的采样点，以致算法性能很差，甚至得不到问题的可行解。
- 种群太大时尽管可增加优化信息以阻止早熟收敛的发生，但无疑会增加计算量，从而使收敛时间太长。
- 当然，在优化过程中种群数目是允许变化的，也可以将较大规模的种群分解成若干子种群进行进化。

交叉概率C

- 交叉概率用于控制交叉操作的频率。
 - 标准GA中每一代有 $N \times C$ 个个体进行交叉
- 交叉概率太大时，种群中串的更新很快，进而会使高适配值的个体很快被破坏掉。
- 概率太小时，交叉操作很少进行，从而会使搜索停滞不前。

变异概率M

- 变异概率是加大种群多样性的重要因素。
- 基于二进制编码的GA中，每一代有 $N*M*L$ 个基因发生变异(为码长)，通常一个较低的变异率足以防止整个群体中任一位置的基因一直保持不变。
- 变异概率太小则不会产生新个体。
- 变异概率太大则使GA成为随机搜索。

代沟G

- 代沟用于控制每代中种群被替换的比例，即每代有 $N \cdot (1-G)$ 个父代个体被选中进入下一代种群。
 - $G=100\%$ 意味着所有个体将被替换
 - $G=50\%$ 意味着一半种群将被替换
- 标准GA中 $G=100\%$
- 有些替换策略中 G 值跟新旧个体的适配值好坏有关而且是变化的，如将父代种群的临时种群择优构成下一代种群。

尺度窗口W

- 该参数用于作出由目标值到适配值的调整。
 - 譬如最大化函数优化问题，个体 x 的适配值可以定义为 $u(x) = f(x) - f_{\min}$ ，其中 $f(x)$ 为其目标值、 f_{\min} 为问题的最小目标值。然而，通常 f_{\min} 是未知的，许多方法用算法当前进程发现的最小目标值 f'_{\min} 作为替代，但显然定义不同的 f'_{\min} 将影响优良个体的选择压力。
- Grefenstette(1986)定义了3种方案：
 - 若 $W=0$ ，则首先令 f'_{\min} 为第1代种群的最小目标值，在后续的进化中忽视选择目标小于 f'_{\min} 的个体，而当种群中所有个体目标大于 f'_{\min} 时才更新；
 - 若 $0 < W < 7$ ，则令 f'_{\min} 为发生在最近 W 代进化中的最小目标值；
 - 若 $W=7$ ，则不做尺度变换操作，即无穷大窗口。

选择策略S

- 通常有两种选择策略：
 - 其一为纯选择(pure selection)，记 $S=P$ ，即种群中每一个体根据其适配值作比例选择；
 - 其二为保优策略(elitist strategy)，记 $S=E$ ，即先用纯选择进行选择，然后将best so far加入下一代种群，该策略可防止最优解的遗失。

推荐参数

- De Jong(1975)认为函数优化的最佳SGA参数为：
 - 种群数50
 - 交叉概率0.6
 - 变异概率0.001
 - 代沟100%
 - 尺度窗口无穷大
 - 采用保优策略
- Grefenstette(1986)则将GA的参数选取作为一个优化问题，提出用GA优化GA参数的二级数值方法。他认为最佳在线GA参数和离线GA参数(括号内数值)为：
 - 种群数30(80)
 - 交叉概率0.95(0.45)
 - 变异概率0.01(0.01)
 - 代沟100%(90%)
 - 尺度窗口1(1)
 - 保优(不保优)
 - 尽管此方法适用范围较广，但工作量较大，且二级算法本身的参数也有待优化，因此很少得到实际应用。

评论

- 显然最佳GA参数是依赖于问题的，而上述结论仅针对标准GA的函数优化。
- 尽管GA是一种有效的优化算法，但其最优参数的确定本身就是一个极其复杂的优化问题。
- 王凌等将最佳GA参数的确定问题描述为一个随机优化问题，进而利用序优化的思想和最优计算量分配技术来确定最佳参数组合。鉴于该方法论的指导，可以用较少的仿真计算量来以较大的置信度确定适合于所研究问题的最优参数，并且适用于任何类型的优化问题。
- 需要指出的是，由于GA是一个动态寻优过程，一成不变的控制参数未必合理，自适应控制参数有助于调整算法的搜索行为和能力，譬如在进化初期采用较大的遗传概率使得种群有足够的多样性，有助于全局搜索，而在进化后期采用较小的遗传概率来增强局部搜索能力和加速收敛。该问题的解决将有利于GA理论和应用的发展，因此研究最优参数的选取和控制仍旧是GA研究的重要内容之一，尤其是理论研究。

遗传操作

- 种群初始化
- 选择
- 交叉
- 变异
- 种群替换

种群初始化

- 鉴于保优GA的概率1收敛特性，初始种群通常是随机产生的。
- 考虑到搜索效率和质量：
 - 一方面要求尽量使初始种群分散地分布于解空间
 - 另一方面可以采用一些简单方法或规则快速产生一些解作为GA的初始个体
 - 如优化flow shop问题时，可以用NEH方法产生一个GA初始个体，其余个体则在解空间中均匀选取

选择操作

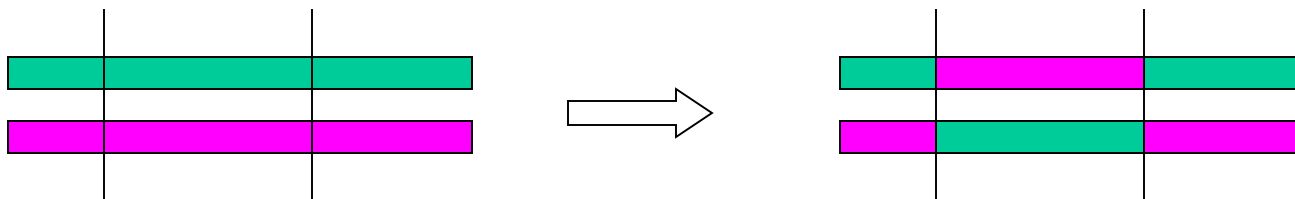
- 选择操作用于避免有效基因的损失，使高性能的个体得以更大的概率生存，从而提高全局收敛性和计算效率。最常用的方法是比例选择、基于排名的选择和锦标赛选择。
 - 比例选择(轮盘赌)：用正比于个体适配值的概率来选择相应的个体，即产生随机数 $\xi \in [0,1]$ ，若下式成立则选择状态i进行复制，其中 f_i 为个体i的适配值。

$$\sum_{j=1}^{i-1} f_j / \sum_{j=1}^{Pop_size} f_j < \xi \leq \sum_{j=1}^i f_j / \sum_{j=1}^{Pop_size} f_j$$

- 基于排名的选择：首先将种群中所有个体由好到坏进行排列，然后以一定方式分配给各个体一定的选择概率，具体分配方式不限，如线性方式、非线性方式，但要求越好的个体所分配的概率越大，且所有个体所分配的概率之和为1。
 - 锦标赛选择(tournament selection)：首先在父代种群中随机选取个个体，然后令其中适配值或目标值最好的个体为被选中个体。显然，的大小影响选择性能。

二进制和十进制编码的交叉操作

- 交叉操作作用于组合出新的个体，在解空间中进行有效搜索，同时降低对有效模式的破坏概率。
- 二进制编码GA通常采用单点交叉和多点交叉。
 - 单点交叉：首先随机确定一个交叉位置，然后对换交叉点后的子串。譬如，父串为(1 0 1 1|0 0 1)和(0 0 1 0|1 1 0)，若单点交叉位置为4，则后代个体为(1 0 1 1|1 1 0)和(0 0 1 0|0 0 1)。
 - 多点交叉：首先随机确定多个交叉位置，然后对换相应的子串。若两点交叉，交叉位置为2、5，父代个体同上，则后代个体为(1 0|1 0 1|0 1)和(0 0|1 1 0|1 0)。



- 十进制编码GA的交叉操作类似于二进制编码GA。

实数编码的交叉操作

- 实数编码GA通常采用双个体算术交叉或多个个体算术交叉。
- 双个体算术交叉：

针对选中的两个个体进行如下交叉，即 $x_1' = \alpha x_1 + (1 - \alpha)x_2$ ， $x_2' = \alpha x_2 + (1 - \alpha)x_1$ ，其中随机数 $\alpha \in (0,1)$ ， x_1 ， x_2 为父代个体， x_1' ， x_2' 为后代个体。

- 多个体算术交叉

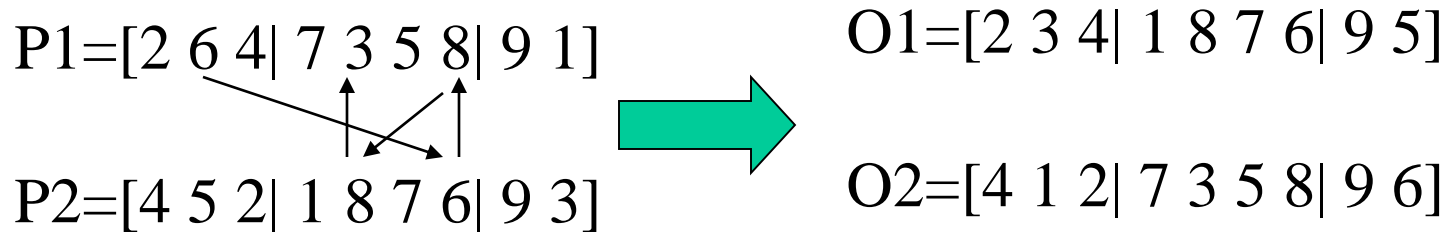
针对多个选中个体进行如下交叉，即 $x' = \alpha_1 x_1 + \cdots + \alpha_n x_n$ ，其中 x_i 为父代个体， x' 为后代个体， $\alpha_i \in [0, 1]$ 且 $\sum_{i=1}^n \alpha_i = 1$ 。

置换编码的交叉操作

- 部分映射交叉
- 次序交叉
- 循环交叉
- 基于位置的交叉
- Non-ABEL群交叉
- ...

部分映射交叉(partially mapping crossover, PMX)

- 首先随机选取两个交叉点
- 然后交换父代个体交叉点之间的片段
- 对于交叉点外的基因，若它不与换过来的片段冲突则保留，若冲突则通过部分映射来确定直到没有冲突的基因，从而获得后代个体。
- 譬如：
 - 两个父代个体为 $P1=[2\ 6\ 4\ |\ 7\ 3\ 5\ 8\ |\ 9\ 1]$ ， $P2=[4\ 5\ 2\ |\ 1\ 8\ 7\ 6\ |\ 9\ 3]$
 - 若交叉位置为3、7，则：片段(7 3 5 8)和(1 8 7 6)将交换
 - 对于的剩余基因，由于2不与(1 8 7 6)冲突则直接填入，6存在冲突，6的映射基因为8仍存在冲突，8的映射基因为3不存在冲突，则将3填入后代个体的相应位置

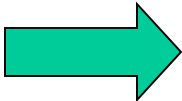


- PMX算子一定程度上满足Holland图式定理的基本性质，子串能够继承父串的有效模式。

Non-ABEL方法

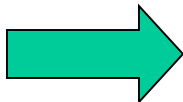
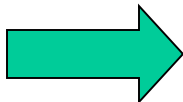
- Non-ABEL方法采用如下公式得到后代个体：
 $q1[i]=p1[p2[i]]$, $q2[i]=p2[p1[i]]$.

P1=[2 6 4 7 3 5 8 9 1] O1=[7 3 6 2 9 8 5 1 4]
P2=[4 5 2 1 8 7 6 9 3] O2=[5 7 1 6 2 8 9 3 4]



次序交叉(order crossover, OX)

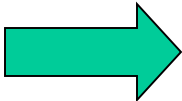
- 与PMX非常类似，它首先随机确定两个交叉位置，并交换交叉点之间的片段，并从第2交叉位置起在原先父代个体中删除将从另一父代个体交换过来的基因，然后从第2交叉位置后开始填入剩余基因。

P1=[2 6 4| 7 3 5 8| 9 1]  O1=[4 3 5| 1 8 7 6| 9 2]
P2=[4 5 2| 1 8 7 6| 9 3]  O2=[2 1 6| 7 3 5 8| 9 4]

单位置次序交叉(Reeves表其为C1)

- 随机产生一个交叉点，保留交叉点前的片段，后余片段则从另一父代个体中剔除已选基因后依次填入。

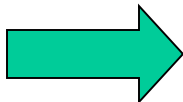
P1=[2 6 4 7| 3 5 8 9 1] O1=[2 6 4 7|5 1 8 9 3]
P2=[4 5 2 1| 8 7 6 9 3] O2=[4 5 2 1|6 7 3 8 9]



线性次序交叉(linear order crossover, LOX)

- 尽管Croce等(1995)用穴(hole)移动的方式来描述LOX，其实它就是双位置次序交叉，与OX相比仅填入基因起始位置不同。
- 具体而言，首先随机确定两个交叉位置，并交换交叉点之间的片段，并在原先父代个体中删除将从另一父代个体交换过来的基因，然后从第1个基因位置起依次在两交叉位置外填入剩余基因。
- 譬如父代个体和交叉点同上，则片段(7 3 5 8)和(1 8 7 6)将交换，在p1中删除(1 8 7 6)后剩余(2 4 3 5 9)，然后将其填入q1就得到后代个体为q1=[2 4 3| 1 8 7 6| 5 9]，相应地可得到个体q2=[4 2 1| 7 3 5 8| 6 9]。

P1=[2 6 4| 7 3 5 8| 9 1]
P2=[4 5 2| 1 8 7 6| 9 3]

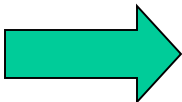


O1=[2 4 3| 1 8 7 6| 5 9]
O2=[4 2 1| 7 3 5 8| 6 9]

基于位置的交叉(position-based crossover, PX)

- PX与OX类似，只是它不再选取连续的基因片段，而是随机选取一些位置，然后交换被选中位置上的基因，并在原先父代个体中删除将从另一父代个体交换过来的基因，然后从第1个基因位置起依次在未选中位置填入剩余基因。
- 譬如，父代个体同前，假设随机选取的位置点为2，3，6，8，则：

P1=[2 6 4 7 3 5 8 9 1] O1=[6 5 2 4 3 7 8 9 1]
P2=[4 5 2 1 8 7 6 9 3] O2=[2 6 4 1 8 5 7 9 3]



循环交叉(cycle crossover, CX)

- 循环交叉将另一个父代个体作为参照以对当前父代个体中的位置进行重组，先与另一父代个体实现一个循环链，并将对应的位置填入相应的位置，如[2 # 4 # # # # # #]和[4 # 2 # # # # # #]，循环组成后再将另一个父代个体的位置填入相同的位置。

P1=[2 6 4 7 3 5 8 9 1]

P2=[4 5 2 1 8 7 6 9 3]

非自循环链1: 2-4-2

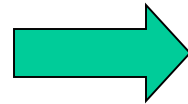
非循环链2: 6-5-7-1-3-8-6

自循环链: 9-9

保留非自循环链1: 2-4-2

P1=[2 6 4 7 3 5 8 9 1]

P2=[4 5 2 1 8 7 6 9 3]



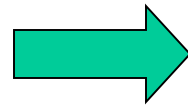
O1=[2 5 4 1 8 7 6 9 3]

O2=[4 6 2 7 3 5 8 9 1]

保留非自循环链2: 6-5-7-1-3-8-6

P1=[2 6 4 7 3 5 8 9 1]

P2=[4 5 2 1 8 7 6 9 3]



O1=[4 6 2 7 3 5 8 9 1]

O2=[2 5 4 1 8 7 6 9 3]

另例

P1=[1 2 3 4 5 6 7 8 9 10]

P2=[1 7 2 5 4 9 3 6 10 8]

非自循环链1: 2-7-3-2

非循环链2: 4-5-4

非循环链3: 6-9-10-8-6

自循环链: 1-1

保留非自循环链1: 2-7-3-2

O1=[1 2 3 5 4 9 7 6 10 8]

O2=[1 7 2 4 5 6 3 8 9 10]

保留非自循环链1和2

O1=[1 2 3 4 5 9 7 6 10 8]

O2=[1 7 2 5 4 6 3 8 9 10]

保留非自循环链2: 6-5-7-1-3-8-6

O1=[1 7 2 4 5 9 3 6 10 8]

O2=[1 2 3 5 4 6 7 8 9 10]

保留非自循环链1和3

O1=[1 2 3 5 4 6 7 8 9 10]

O2=[1 7 2 4 5 9 3 6 10 8]

保留非自循环链3: 6-9-10-8-6

O1=[1 7 2 5 4 6 3 8 9 10]

O2=[1 2 3 4 5 9 7 6 10 8]

保留非自循环链2和3

O1=[1 7 2 4 5 6 3 8 9 10]

O2=[1 2 3 5 4 9 7 6 10 8]

均保留或不保留非自循环链1、2和3

O1=[1 2 3 4 5 6 7 8 9 10]

O2=[1 7 2 5 4 9 3 6 10 8]

变异操作

- 当交叉操作产生的后代适配值不再进化且没有达到最优时，就意味着算法的早熟收敛。这种现象的根源在于有效基因的缺损，变异操作一定程度上克服了这种情况，有利于增加种群的多样性。
- 二进制或十进制编码中通常采用单位位置或多位置替换式变异，即用另一种基因替换某一位置或某些位置上原先的基因。
- 实数编码中通常采用扰动式变异，即对原先个体附加一定机制的扰动来实现变异，即 $x' = x + \eta \cdot \xi$ ，其中 x' 和 x 分别为新旧个体， η 为扰动幅度参数， ξ 为随机扰动变量。 ξ 可以服从高斯分布、柯西分布、均匀分布，也可以为混沌变量或梯度信息。

变异操作(续)

- 组合优化问题中的置换编码GA通常采用互换、逆序和插入变异。
 - 互换操作(SWAP)，即随机交换染色体中两不同基因的位置。
 - 逆序操作(INV)，即将染色体中两不同随机位置间的基因串逆序。
 - 插入操作(INS)，即随机选择某个点插入到串中的不同随机位置。
- 譬如状态为(5 4 1 7 9 8 6 2 3)，两随机位置为2、6，则：
 - SWAP的结果为(5 8 1 7 9 4 6 2 3)
 - INV的结果为(5 8 9 7 1 4 6 2 3)
 - INS的结果为(5 8 4 1 7 9 6 2 3)

替换策略

- 种群替换策略通常采用整体替换或部分替换。
 - 整体替换即将新种群完全覆盖原先种群
 - 部分替换则用部分新个体替换部分旧个体
 - 譬如(μ , λ)策略由 μ 个父代个体产生 λ 个后代个体，然后从后代个体中选取 μ 个最好的个体作为下一代种群
 - ($\mu + \lambda$)策略则从所有 μ 个父代个体和 λ 个后代个体中选取最好的 μ 个个体作为下一代种群

终止条件

- GA的收敛理论说明了GA具有概率1收敛的极限性质，然而实际算法通常难以实现理论上的收敛条件。
- 因此，追寻的目标应该是具有一定优化质量的快速搜索，这显然与算法操作设计和参数选取有关。
- 由于实际应用GA时不允许让其无停止地进行搜索，同时问题的最优解也通常未知，因此必须设计一些近似收敛准则来终止算法。常用方法包括：
 - 给定一个最大进化步数
 - 给定个体评价总数
 - 给定最佳搜索解的最大滞留步数等。
- GA是一种复杂的非线性随机智能优化计算模型，纯粹用数学方法来预测其运算结果很难，目前为兼顾优化质量和效率所采取的设计方法大多是经验法或试探法，该方面理论有待更深入的研究与完善。计算机仿真和数学分析相结合的研究途径比较可取。