演化规划概述

姓名：葛丛丛　　学号：11721071

May 4, 2018

目录

1 演化规划简介

1.1 概念与背景

演化计算借鉴生物界自然选择法则、遗传机制,利用选择、交叉、变异等操作模拟生物进化的过程以解决实值连续函数全局优化、神经网络结构优化、模式识别与系统辨识等问题[1, 2, 3]。其中，演化规划是演化计算中的代表方法之一，演化规划是由L.J.Fogel等在20世纪60年代提出的。当时演化规划的目标是通过模拟进化来获得智能行为。他们将智能视为能够预测其所在环境的状态，并按照预定目标作出适当响应的能力。对环境的预测能力是智能行为的一个重要特征。演化规划模拟生物种群层次上的进化，因此在进化过程中主要强调生物种群行为上的联系，即强调种群层次上的行为进化而建立父、子代间的行为链。

1.2 演化规划的基本结构

Fogel将环境描述为由有限字符集中的符号所组成的序列，而预测器则用有限状态自动机来表示。一个有限状态自动机是一个五元组(S, I, O, δ, s_0)。其中S是状态的集合，I 是输入符号的集合，O是输出符号的集合，$\delta : S \times I \to S \times O$是转移函数，$s_0 \in S$是初始状态。图6.1给出了有限状态机的一个简单的例子。

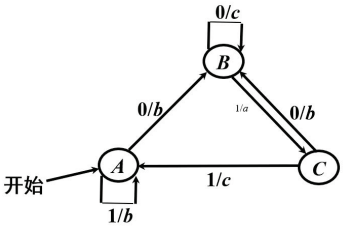

图 1: 一个有限状态自动机

在图1所示的有限状态自动机中，$S = \{A, B, C\}, I = \{0, 1\}, O = a, b, c$，两个状态之间的一条有向边指示一个状态转移，而状态转移函数$\delta : S \times I \to S \times O$由边上形如的标记所指明。譬如，从状态A到状态B之间的有向边的标记为$0/b$，则该标记所表示的状态转移为$\delta((A, 0) = (B, b))$，即若当前状态为A且输入符号为0时，机器转移到状态B且输出符号b。初始状态为A。

演化规划就是通过模拟生物进化的方式演化出能够执行预测任务的有限状态自动机。一个简单的预测任务是：给定一个序列$a_1, a_2, ...$，在观察到前n个符号$a_1, a_2, ..., a_n$的基础上，预测第$n + 1$个符号。当输入序列为$a_1, a_2, ..., a_n$时，有限状态自动机产生一个输出序列$b_1, b_2, ..., b_n$，其中$b_i (i = 1, 2, ..., n - 1)$是对$a_{i+1}$的预测。一个执行这种预测任务的自动机如图2所示。例如，当输入序列为011101时，所产生的输出序列为110111。这时，当n=1，2，5时，机器作出了准确的预测，预测准确率为60%。

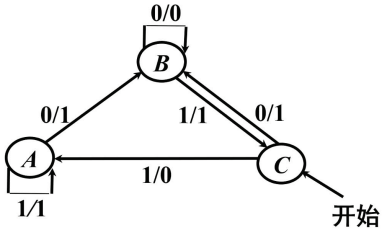

图 2: 一个有限状态自动机

用演化规划求解上述问题的方法是：保持一个具有μ个有限状态自动机的种群，对种群中的每个自动机进行变异得到μ 个后代。变异通常有改变输出符号、改变状态转移、添加一个状态、删除一个状态和改变初始状态五种方式。然后根据对有限状态自动机的某种适应值度量，从μ个父体和μ个后代中选取μ个个体作为下一代种群。算法流程如算法1所示。

2 演化规划的实现技术

演化规划分为标准演化规划、元演化规划和旋转演化规划三种，它们对应的表示方法如下所示：
(1)标准演化规划$x = (x_1, x_2, ..., x_n)$

Algorithm 1: Procedure Evolutionary Programming

```
1  begin
2  t ← 0
3  initialize(P(t) = a_1(t), a_2(t), ..., a_μ(t));
4  evaluate(P(t));
5  while not termination condition do
6  |  for i ← 1 to μ do
7  |  |  O_i(t) ← mutate(a_i(t));
8  |  |  evaluate(o_i(t));
9  |  end
10 |  P(t+1) ← survivor_select(P(t) ∪ o_1(t), ..., o_μ(t));
11 |  t ← t + 1;
12 end
13 return the best solution;
```

(2)元演化规划$(x,\sigma)=(x_1,x_2,...,x_n,\sigma_1,\sigma_2,...,\sigma_n)$

(3)旋转演化规划$(x,\sigma,\rho)=(x_1,x_2,...,x_n,\sigma_1,\sigma_2,...,\sigma_n,\rho_1,\rho_2,...,\rho_m)$

其中，$m=n(n-1)/2$，ρ_1表示x_1与x_2之间的相关系数，ρ_2表示x_1与x_3之间的相关系数，... ，ρ_m表示x_{n-1}与x_n之间的相关系数。若ρ_k表示变量x_i与x_j之间的相关系数，则x_i与x_j之间的协方差由下式确定:

$$\rho_k=\frac{c_{ij}}{\sqrt{\sigma_i\sigma_j}},\ \rho_k\in[-1,1] \tag{1}$$

由协方差$c_{ij}(i=1,2,...,n-1;j=i+1,...,n)$可以构成协方差矩阵C

$$C=\begin{bmatrix} \sigma_1 & c_{12} & ... & c_{1n} \\ c_{21} & \sigma_2 & ... & c_{2n} \\ ... & ... & ... & ... \\ c_{n1} & c_{n2} & ... & \sigma_n \end{bmatrix} \tag{2}$$

其中，$c_{ij}=c_{ji}(i\neq j)$，C用于产生服从n维正态分布的随机向量$N(0,C)$。

2.1 变异操作

变异操作：在演化规划算法中，必须要考虑变异操作。应该指出，演化规划中没有重组或交换这类算子，它的进化主要依赖突变。在标准演化规划中这类突变十分简单，它只需要参照个体适应度添加一个随机数即可。很明显，标准演化规划在演化过程中的自适应调整功能主要依靠适应度$F(x)$来实现。

对于标准演化规划而言，个体可表示为$x=(x_1,x_2,...,x_n)$，对应的变异操作可由如下公式表示：

$$\begin{cases} x_i=x_i+\sigma_i\cdot N_i(0,1) \\ \sigma_i=\sqrt{\beta\cdot F(x)+\gamma_i} \end{cases} \tag{3}$$

其中$F(x)$为个体x的适应值，$\beta_i,\gamma_i(i=1,2,...,n)$为待定的参数。通常取$\beta_i=1$，$\gamma_i=0$。

为了增加演化规划在进化过程中的自适应调整功能，人们在突变中添加方差的概念。对于元演化规划而言，这时个体可表示为$(x,\sigma=(x_1,x_2,...,x_n,\sigma_1,\sigma_2,...,\sigma_n)$，对应的变异操作可由如下公式表示：

$$\begin{cases} x_i^{'}=x_i+\sigma_i\cdot N_i(0,1) \\ \sigma_i^{'}=\sqrt{\eta\cdot\sigma_i}\cdot N_i(0,1) \end{cases} \tag{4}$$

其中η为常系数，σ_i为旧个体第i个分量的标准差，$\sigma_i^{'}$为新个体第i个分量的标准差，$N(0,1)$是针对第i个分量发生的随机数，它服从标准正态分布。

从上式可以看出，新个体也是在旧个体的基础上添加一个随机数，该添加量取决于个体的方差，而方差在每次进化中又有自适应调整。这种演化方式已经成为演化规划的主要手段，因此在演化规划前冠以“元”这个术语以表示它为基本方法。元演化规划（Meta EP）的突变尽管类似于演化策略，但是它们有下述区别：

(1)执行顺序不同。演化规划中首先计算新个体的目标变量$x_i^{'}$，计算中沿用旧个体的标准差σ_i；其次才计算新个体的标准差$\sigma_i^{'}$，新的标准差留待下次进化时才用。与之相反，演化策略是先调整标准差σ，然后再用新的标准差$\sigma^{'}$去更改个体的目标变量X。

(2)计算式的不同。演化规划的计算式要比演化策略的计算式简单。

旋转演化规划（Rmeta EP）进一步扩展演化规划，在表达个体时添加第三个因子——协方差，用三元组表示个体，即$(x,\sigma,\rho=(x_1,x_2,...,x_n,\sigma_1,\sigma_2,...,\sigma_n,\rho_1,\rho_2,...,\rho_m)$，对应的变异操作可由如下公式表示：

$$\begin{cases} x_i^{'} = x_i + N_i(0,C) \\ \sigma_i^{'} = \sigma_i + \sqrt{\eta \cdot \sigma_i} \cdot N_i(0,1) \\ \rho_j^{'} = \rho_j + \sqrt{\xi \cdot \rho_j} \cdot N_j(0,1) \end{cases} \tag{5}$$

其中η,ξ为常系数，x是旧个体的目标变量，其内含n个分量；$x^{'}$是新个体的目标变量，其内含n个分量。$N(0,C)$遵从正态分布的随机数，其数学期望为0,其标准差与协方差有关；ρ_j是相关系数$\rho_j = \frac{c_{ij}}{\sqrt{\sigma_i \sigma_j}}$。

目前已经提出多种变异算子。这些变异算子的区别主要在于：

(1)修改变异步长公式的不同；

(2)在公式中使用方差而不是标准差；

(3)σ和x被变异的次序不同。譬如，对元演化规划，有人提出下面的变异公式：

$$\begin{cases} \sigma_i^{'} = \sigma_i \cdot (1 + \alpha \cdot N_i(0,1)) \quad \alpha \approx 0.2 \\ x_i^{'} = x_i + \sigma_i^{'} \cdot N_i(0,1) \end{cases} \tag{6}$$

2.2 父体选择

父体选择：除了变异算子，演化规划中的父体选择非常简单。在演化规划中，种群中的每个个体经过变异恰好产生一个后代。种群中的每个个体都是一个父体，无需进行专门选择。通常而言，演化规划采用随机型竞争选择，即$q-$竞争选择。在这种方法中，对每个个体$\alpha \in P(t) \cup P_{'}(t)$其中$P_{'}(t)$为$\mu$个后代的集合，从$P(t) \cup P_{'}(t)$中随机地选取$q$个个体。然后将个体$a$的适应值分别与这$q$个个体的适应值进行比较，并记录个体$a$的适应值优于或等于所比较个体适应值的次数，该次数称为个体a的得分。最后，将$P(t) \cup P_{'}(t)$中的个个体按照它们的得分按降序排序，并选择前μ个个体作为下一代种群。总的来说，优良个体进入下一代种群的机会较大，但较差个体也有进入下一代种群的机会。随着q值的增加，较差个体进入下一代种群的机会减小。

以下给出演化规划的一个应用实例，我们考虑Ackley函数的优化问题，如下所示：

$$\min \mathrm{f}(\mathrm{x}_1,\mathrm{x}_2,...,\mathrm{x}_3 0) = -20 \cdot \exp\left(-0.2\sqrt{\frac{1}{30}\sum_{\mathrm{i}=1}^{30}\mathrm{x}_\mathrm{i}^2}\right) - \exp\left(\frac{1}{30}\sum_{\mathrm{i}=1}^{30}\cos(2\pi \cdot \mathrm{x}_\mathrm{i})\right) + 20 + \mathrm{e} \tag{7}$$
$$-30 \le x_i \le 30,\ i = 1,2,...,30;\ e = 2.71282$$

用元演化规划求解该问题，其设计如下：

(1)表示：个体可表示为形如$(x_i,x_2,...,x_3 0,\sigma_1,\sigma_2,...,\sigma_{30})$的形式

(2)适应函数：适应函数取为目标函数。

(3)参数设置：$\mu = 200,\ q = 10,\ \eta = 6$

(4)终止准则：当进行200000次函数值计算或发现最优解后终止算法。

(5)种群初始化：初始种群中每个个体的变量部分随机地产生，每个变量均匀地分布在区间$[-30,30]$内。每个个体的变异步长都相同，设为$\sigma = 3$。运行上述算法10 次，每次找到的最好解都位于全局最优峰上。最好解的平均函数值为$1.39 \cdot 10^{-2}$

3 演化规划的缺陷与改进方法

3.1 演化规划的缺陷

大量的研究证明，由于过度选择、变异操作破坏有效模式、参数选取不当等原因，该算法存在一些亟待克服的缺点：(1)容易出现早熟收敛现象，从而陷入局部极值点；(2) 进化后期，个体之间的竞争趋缓导致算法后期的搜索效率降低；(3)对初始参数敏感。在传统EP算法中, 参与进化的只有一个种群，

因此在这个唯一的种群中的个体只能遵循唯一的规律进行进化(相同的高斯变异算子标准差σ_0)。较大的高斯变异算子可以保证种群具有较好的探索能力，即能够以比较大的概率到达解空间的各个地方，种群(或者种群中的部分个体)能够以比较大的概率落在全局最优解所在的邻域, 在每一个局部极值都具有很好的局部逃逸能力。但此种探索(局部逃逸)能力的取得是以降低解的精度为代价的，种群总是在全局最优解的附近跳转，无法使用足够小的变异来以足够高的精度逼近最优解。较小的高斯变异算子能够保证种群在局部具有良好的搜索能力，即能够以高的精度找到种群所在局部的极值，但小的变异算子不能(或者需要很多的进化次数才能)产生足够大的变异从而使种群从这个局部进化到具有更大极值(或者全局最优解)的局部，从而早熟收敛. 由于我们无法预知函数各个局部极值(包括全局最优解)的位置, 也无法预知局部极值间的距离以及离全局最优解最近的局部极值与全局最优解间的距离，因此无法选取合适的σ，使其既能大到足够使个体产生大的变异从局部极值点逃逸, 又能小到使个体产生小的变异, 在全局最优解附近以高的精度逼近最优解。

3.2 改进方法与研究进展

近年来, 一些学者提出了很多方法来改善EP算法的性能[4, 5]，文献[4]提出了一种基于Lévy概率分布的进化规划算法LEP，在该算法中使用Lévy变异代替变异, 其核心思想是使用具有更大方差的变异算子来促使种群更快地收敛到函数的最优解。但使用大方差的变异算子将使种群逼近最优解的速度变慢，精度降低。文献[5]提出了一种并行进化算法(Parallel Evolutionary Programming，PEP)，在该算法中，多个种群同时使用高斯变异算子和柯西变异算子，其核心思想是同时利用高斯变异算子良好的局部搜索能力和柯西变异算子良好的全局探索能力，以期既能快速收敛到最优解附近，又能有比较好的逼近精度。但是同一个体使用两种变异方式，导致算法的计算复杂度增加，算法的收敛速度受到影响，同时该算法对初始参数敏感。文献[6]提出了一种新的双群进化规划算法，改进了上述算法的缺陷。在该算法中，进化在两个不同的子群间并行进行，通过使用不同的变异策略，实现种群在解空间具有尽可能分散的探索能力的同时在局部具有尽可能细致的搜索能力，通过子群重组实现子群间的信息交换。

参考文献

[1] S. Bornholdt and D. Graudenz, "General asymmetric neural networks and structure design by genetic algorithms," *Neural networks*, vol. 5, no. 2, pp. 327–334, 1992.

[2] A. V. Sebald and J. Schlenzig, "Minimax design of neural net controllers for highly uncertain plants," *IEEE Transactions on Neural Networks*, vol. 5, no. 1, pp. 73–82, 1994.

[3] X. Yao and Y. Liu, "A new evolutionary system for evolving artificial neural networks," *IEEE transactions on neural networks*, vol. 8, no. 3, pp. 694–713, 1997.

[4] C.-Y. Lee and X. Yao, "Evolutionary programming using mutations based on the lévy probability distribution," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 1, pp. 1–13, 2004.

[5] S. Tongchim and X. Yao, "Parallel evolutionary programming," in *Evolutionary Computation, 2004. CEC2004. Congress on*, vol. 2, pp. 1362–1367, IEEE, 2004.

[6] X. Wang, D. Xiang, T. Jiang, C.-S. Lin, S.-G. Gong, and X. Fang, "A novel bi-group evolutionary programming," *CHINESE JOURNAL OF COMPUTERS-CHINESE EDITION-*, vol. 29, no. 5, p. 835, 2006.