migrate 参考资料

概要: 子种群间迁移个体。

描述:

该函数在当前种群 Chrom 的子种群之间实现个体迁移，并返回迁移后的种群 Chrom。

语法:

Chrom = migrate(Chrom, SUBPOP)

Chrom = migrate(Chrom, SUBPOP, MIGR)

Chrom = migrate(Chrom, SUBPOP, MIGR, Select)

Chrom = migrate(Chrom, SUBPOP, MIGR, Select, Structure)

Chrom = migrate(Chrom, SUBPOP, MIGR, Select, Structure, FitnV)

[Chrom, ObjV] = migrate(Chrom, SUBPOP, MIGR, Select, Structure, FitnV, ObjV)

[Chrom, ObjV, LegV] = migrate(Chrom, SUBPOP, MIGR, Select, Structure, FitnV, ObjV, LegV)

详细说明:

Geatpy 中子种群的染色体矩阵是基于种群染色体矩阵 Chrom 的，子种群之间拥有相同数量的个体，并且按照下列方案进行有序排列：

$$\text{Chrom} = \begin{pmatrix} subchrom_1\_ind_1 \\ subchrom_1\_ind_2 \\ \vdots \\ subchrom_1\_ind_n \\ subchrom_2\_ind_1 \\ subchrom_2\_ind_2 \\ \vdots \\ subchrom_2\_ind_n \\ \vdots \\ subchrom_{SUBPOP}\_ind_1 \\ subchrom_{SUBPOP}\_ind_2 \\ \vdots \\ subchrom_{SUBPOP}\_ind_n \end{pmatrix}$$

Chrom 是一个种群矩阵，矩阵的每一行代表种群中的一个个体的染色体。本函数没对种群的编码方式作特殊要求。

SUBPOP 是一个正整数，代表子种群的数量。SUBPOP 必须能够被种群个体数整除。

MIGR 是一个在 [0,1] 间的实数，代表种群间的迁移概率。缺省情况下默认为 0.2，即子种群中将会有 20% 的个体参与迁移。

Select 是一个正整数，表示种群迁移时所选择的方式。Select 为 0 时表示均匀迁移，此时会对子种群的个体进行随机排序，然后选出排序较前的一些个体进行迁移；Select 为 1 时表示基于适应度的迁移，此时会根据子种群个体的目标函数值 ObjV 或适应度 FitnV 进行排序，并选出该值较小的一些个体进行迁移。

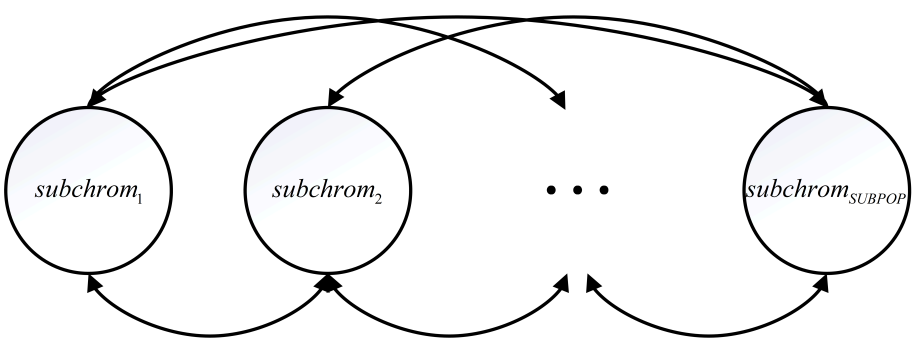
LegV 和 FitnV 都是列向量，LegV 代表种群中各个个体的可行性 (0 为非可行解，1 为可行解)；FitnV 代表种群中各个个体的适应度。

ObjV 是种群个体的目标函数值组成的矩阵，每一列对应一个目标，因此该函数适用于多目标的种群。

Structure 是一个正整数，表示种群迁移的结构。为 0 时表示完全网状结构；为 1 时表示邻近结构；为 2 时表示环状结构。

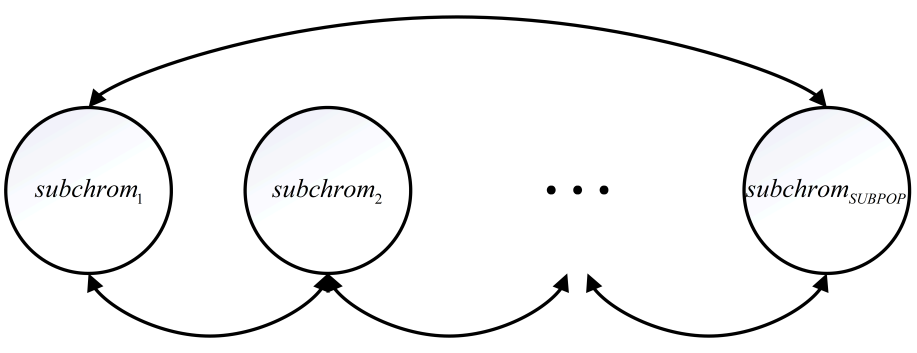
1) 完全网状结构：

每个子种群都会向其他子种群迁移个体。



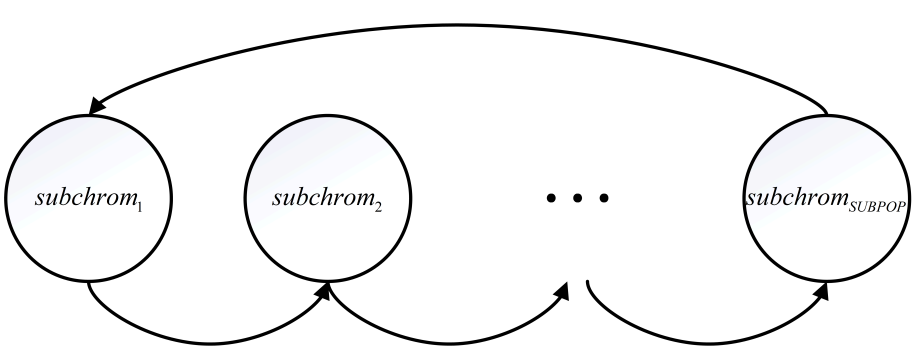
2) 邻近结构：

将子种群首尾相接，相邻的两个子种群之间发生个体迁移。



3) 环状结构：

将子种群首尾相接，1 号子种群的个体向 2 号子种群迁移，2 号子种群个体向 3 号子种群迁移……最后一个子种群的个体向 1 号子种群迁移。



特别注意:

本函数是根据随机方法或根据 FitnV 来进行种群迁移的，与 ObjV 无关，因此在调用本函数前，不需要对传入的 ObjV 乘上’maxormin’(最大最小化标记)，对于返回的 ObjV，也不需要乘上’maxormin’ 进行还原。

应用实例:

现有一个拥有 3 个子种群的种群，其染色体矩阵如下：

$$\text{Chrom} = \begin{pmatrix} 2 & 2 & 2 & 0 \\ 1 & 0 & 2 & 2 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$

对应的个体适应度值分别为：1，2，3，4，5，6。下面在这些子种群中选择 20% 的个体按照其适应度大小，依照邻近结构来进行个体迁移：

```
Chrom = np.array([
   [2,2,2,0],
   [1,0,2,2],
   [0,1,1,0],
   [0,0,1,0],
   [1,1,1,0],
   [0,0,1,1]])
SUBPOP = 3
FintV = np.array([
   [1],
   [2],
   [3],
   [4],
   [5],
   [6]])
Chrom = migrate(Chrom, SUBPOP, 0.2, 1, 0, FintV)
```

迁移后的种群染色体矩阵如下：

$$\text{Chrom} = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & 0 & 2 & 2 \\ 2 & 2 & 2 & 0 \\ 0 & 0 & 1 & 0 \\ 2 & 2 & 2 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$

解析：在邻近结构下，相邻的子种群之间发生个体迁移。假设本例中的 3 个子种群分别是p_1, p_2, p_3，则对比迁移前后的种群染色体可见，a2 的个体迁移到 a1 后，被 a3 迁移到 a1 的个体覆盖掉了。同样地，a2 迁移到 a3 的个体也被 a1 迁移到 a3 的个体所覆盖。

实际上，完全网状结构以及邻近结构都会出现迁移个体被另一个迁移个体覆盖的情况，而环状结构不会。

参考文献:

[1] H. Mühlenbein, M. Schomisch and J. Born, “The Parallel Genetic Algorithm as a Function Optimizer”, Parallel Computing, No. 17, pp.619-632, 1991.

[2] T. Starkweather, D. Whitley and K. Mathias, “Optimization using Distributed Genetic Algorithms”, In Parallel Problems Solving from Nature, Lecture Notes in Computer Science, Vol. 496, pp. 176-185, Springer, 1991.

[3] R. Tanese, “Distributed Genetic Algorithms”, Proc. ICGA 3, pp. 434-439, Morgan Kaufmann Publishers, 1989.

[4] H.-M. Voigt, J. Born and I. Santibanez-Koref, “Modelling and Simulation of Distributed Evolutionary Search Processes for Function Optimization”, Parallel Problems Solving from Nature, Lecture Notes in Computer Science, Vol. 496, pp. 373-380, Springer Verlag, 1991.