10. 约束优化

堵威

华东理工大学 自动化系

2021.5.6

回顾

–多目标优化

实际优化问题通常包含多个目标，这些目标互相冲突

–Pareto最优性

支配：称$\mathbf{x}^*$支配$\mathbf{x}$，如果下面的两个条件成立：(1) 对所有$i \in [1,k]$，$f_i(\mathbf{x}^*) \le f_i(\mathbf{x})$；（2）对至少一个$j \in [1,k]$，$f_j(\mathbf{x}^*) < f_j(\mathbf{x})$，即对于所有目标函数值，$\mathbf{x}^*$至少与$\mathbf{x}$一样好，并且至少有一个目标函数值比$\mathbf{x}$好，记为$\mathbf{x}^* \prec \mathbf{x}$。

非支配、Pareto最优点、Pareto最优集、Pareto前沿

–评判Pareto解集优劣的指标

超体积HV，反世代距离IGD

回顾

- 非支配排序遗传算法、改进的非支配排序遗传算法
  - Nondominated sorting genetic algorithm-II, NSGA-II
  - 迄今为止最经典的进化多目标优化算法（IEEE Transactions on Evolutionary Computation, 2002, 6(2): 182-197）
  - 针对NSGA进行的3大改进之处：

1）快速非支配排序方法（fast nondominated sorting approach）

2）个体拥挤距离算子（crowding distance-based operator）

3）精英策略选择算子（elitism strategy-based selection operator）

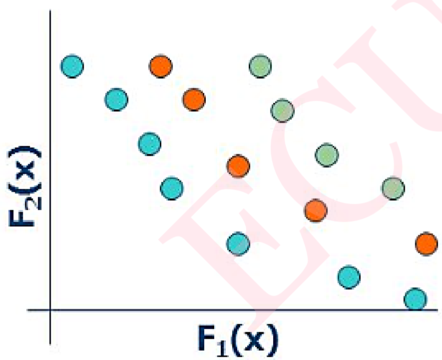
回顾

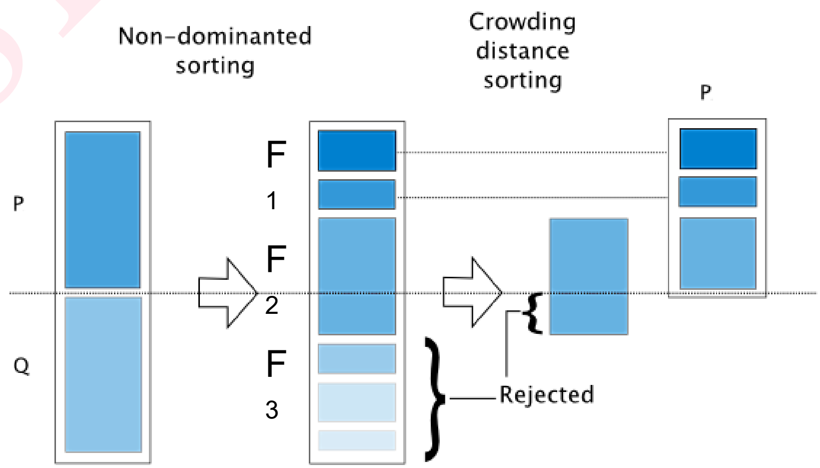
- 改进的非支配排序遗传算法
  - NSGA-II:

1）快速非支配排序方法（fast nondominated sorting approach）

2）个体拥挤距离算子（crowding distance-based operator）

3）精英策略选择算子（elitism strategy-based selection operator）





本章内容

1. 背景知识

2. 罚函数法

3. 处理约束的常用方法

本章内容

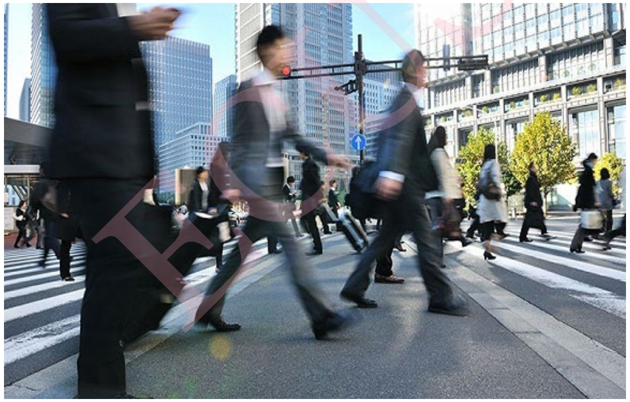
1. 背景知识

2. 罚函数法

3. 处理约束的常用方法

背景知识

- 约束优化
  - 对于实际中的优化问题，基本都会带有约束
  - 如上班怎么选择乘车路线，才能舒服又快速的到达公司？比如约束条件可能包括：总花费、出发和到达时间等
  - 生产线加工订单，如何最快地加工完所有订单？约束条件可能包括：产能限制、生产线匹配等



背景知识

- 约束优化
  - 约束优化问题可以写成：

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) \quad \text{s.t.} \quad \left.\begin{aligned} & g_i(\boldsymbol{x}) \leqslant 0, \quad i \in [1, m], \\ \text{且} \quad & h_j(\boldsymbol{x}) = 0, \quad j \in [1, p]. \end{aligned}\right\}$$

这个问题包含m+p个约束，其中m个不等式约束，p个等式约束。满足所有约束的**x**的集合称为可行集，违反一个或多个约束的**x**的集合被称为不可行集：

$$\text{可行集}\mathcal{F} = \{\boldsymbol{x} : g_i(\boldsymbol{x}) \leqslant 0, \quad i \in [1, m] \text{ 且 } h_j(\boldsymbol{x}) = 0, \quad j \in [1, p]\},$$

$$\text{不可行集}\bar{\mathcal{F}} = \{\boldsymbol{x} : \boldsymbol{x} \notin \mathcal{F}\}.$$

为求解上式优化问题而设计的进化算法被称为约束进化算法。

背景知识

- 约束进化算法
  - 罚函数法：

  基于个体x违反约束的程度修改它的目标函数（适应度值）。

    - 容许甚至鼓励种群中的不可行解的罚函数法被称为外点法，它们会在目标函数（适应度值）或选择上惩罚不可行候选解。
    - 不容许种群中出现不可行解的罚函数法称为内点法。
  - 特殊表示：

  将问题表示为无约束的，但候选解仍然带约束。

  - 修补算法：

  修补不可行的个体让它们变为可行的。

  - 混合方法：

  上述方法的结合，或与非进化算法结合。

本章内容

1. 背景知识

2. 罚函数法

3. 处理约束的常用方法

罚函数法

- 基本概念
  - 罚函数法惩罚违反约束或接近违反约束的候选解
  - 1943年，Richard Courant 首先提出罚函数法
  - 罚函数法是最常用的求解约束优化的方法
  - 两种方法设计罚函数法：

1）内点法：在可行个体靠近约束的边界时就惩罚它，不容许种群中出现不可行个体；

2）外点法：容许种群中存在不可行个体，但在适应度值上惩罚它们，或在为下一代选择父代的时候惩罚它们

罚函数法

- 内点法
  - 内点法在种群中只容许可行个体，当可行个体靠近约束的边界时，会在适应度值上惩罚个体以鼓励个体留在可行范围之内
  - 例：考虑问题

$$\min f(x) = x^2, \quad s.t.\ x \geq c$$

可以修改这个问题，当x的可行值靠近约束时就惩罚它。修改后的函数被称为闸函数。如将该问题转化为无约束问题：

$$\min f'(x) = x^2 + (x-c+\delta)^{-\alpha}$$

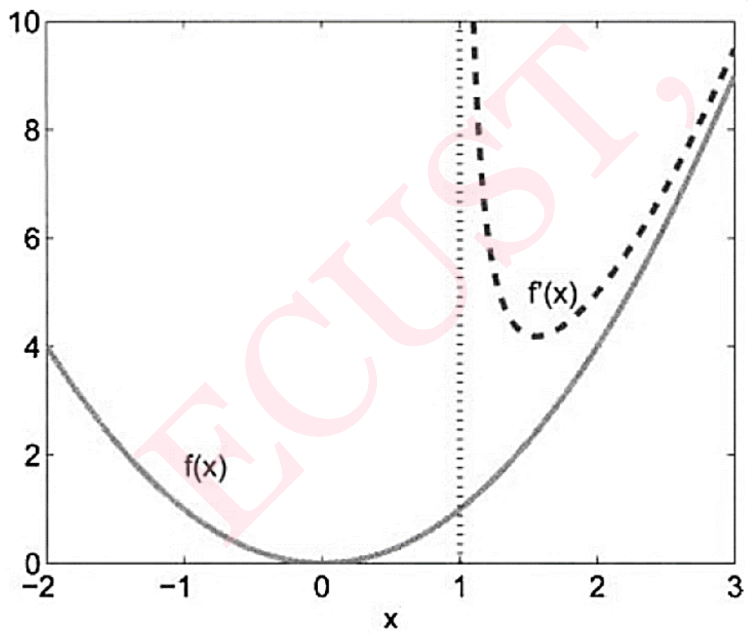
其中，$\delta>0$是一个小的常数，$\alpha>0$是另一个常数，当$\alpha\rightarrow 0$时，$\underset{x}{\operatorname{argmin}} f'(x) \rightarrow \underset{x}{\operatorname{argmin}} f(x)$

罚函数法

- 内点法

$f(x)=x^2$，$f'(x)=x^2+(x-c+\delta)^{-\alpha}$

画出c=1，δ=0.01且α=1时的f(x)和f'(x)



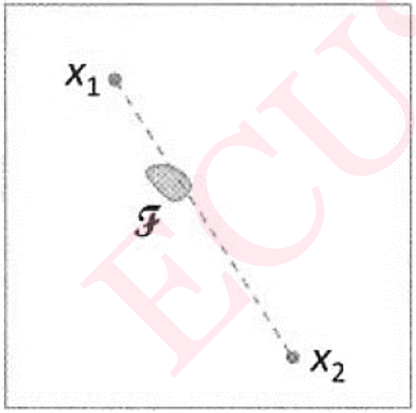
想要最小化f(x)并且x≥1，用内点法将带约束的最小化f(x)转化为无约束的最小化f'(x)

罚函数法

- 内点法
  - 在约束进化算法中，常常不用内点法，原因如下：

1）对于许多约束优化问题，要找出满足所有约束的候选解是一件极具挑战性的问题；

2）不可行解包含的信息可能有利于搜索带约束的最优值



大搜索空间中小可行集*F*的例子。要直接找到候选解$x \in F$ 可能较难，但找到两个不可行个体$\mathbf{x}_1$和$\mathbf{x}_2$，由它们组合生成可行个体也许很容易

罚函数法

- 外点法
  - 外点法容许种群中存在不可行个体，但要在适应度值上或选择概率上惩罚它们
  - 死刑法：死刑法在种群中容许不可行个体，但只是短暂地容许。死刑法会立即去掉种群中的不可行个体x，如果是通过交叉得到的x，就拒绝它并再进行交叉直到得到可行个体；如果是通过变异得到的x，就拒绝它直到得到可行个体
  - 非死刑法：是比死刑法更宽容的外点法，在进化算法的整个过程中都容许不可行个体留在种群中

罚函数法

- 外点法
  - 非死刑法：是比死刑法更宽容的外点法，在进化算法的整个过程中都容许不可行个体留在种群中
  - 将标准约束问题转化为下面的无约束问题：

违反约束的程度

$$\left.\begin{aligned}&\min_{\boldsymbol{x}} \phi(\boldsymbol{x}),\ \text{这里}\ \phi(\boldsymbol{x}) = f(\boldsymbol{x}) + \sum_{i=1}^{m} r_i G_i(\boldsymbol{x}) + \sum_{j=1}^{p} c_j L_j(\boldsymbol{x})\\ &G_i(\boldsymbol{x}) = [\max\{0, g_i(\boldsymbol{x})\}]^{\beta},\\ &L_j(\boldsymbol{x}) = |h_j(\boldsymbol{x})|^{\gamma},\end{aligned}\right\}$$

其中，r_i和c_j为正数，称为惩罚因子；β和γ是正数，通常设为1或2。φ(**x**)通常被称为惩罚适应度函数。如果**x**∈F，则φ(**x**)=f(**x**)；如果**x**∉F，则φ(**x**)>f(**x**)的量会随着违反约束的量的增大而增大。

罚函数法

- 外点法
  - 有了惩罚适应度函数φ(**x**)，就以φ(**x**)作为适应度函数运行优化算法选择下一代个体，因此，可以将无约束优化进化算法扩展到约束优化
  - 约束$h_j(\mathbf{x})=0$很苛刻，如果在连续搜索域上随机生成一个初始种群，所得的个体能满足约束的可能性基本为0，因此常将等式的硬约束转化为要求$h_j(\mathbf{x})$几乎为0而不是正好为0的软约束：

$$|h_j(\mathbf{x})| \le \varepsilon$$

ε为一个小的正数，等价于：

$$h_j(\mathbf{x})-\varepsilon \le 0,\ -h_j(\mathbf{x})-\varepsilon \le 0$$

本章内容

1. 背景知识

2. 罚函数法

3. 处理约束的常用方法

处理约束的常用方法

- 概述
  - 介绍进化算法常用的处理约束的几种方法，都属于非死刑法

1）静态惩罚方法

2）可行点优势

3）折中进化算法

4）协同进化惩罚

5）动态惩罚方法

6）自适应惩罚方法

7）自身自适应罚函数

8）随机排名

9）小生境惩罚方法

....

处理约束的常用方法

- 静态惩罚方法

  – 将等式约束转换为不等式约束：

$$\min_{\boldsymbol{x}} \phi(\boldsymbol{x}),\ 这里\ \phi(\boldsymbol{x}) = f(\boldsymbol{x}) + \sum_{i=1}^{m+p} r_i G_i(x)$$

$$G_i(\boldsymbol{x}) = \begin{cases} (\max\{0, g_i(\boldsymbol{x})\})^\beta, & i \in [1, m], \\ (\max\{0, |h_i(\boldsymbol{x})| - \epsilon\})^\beta, & i \in [m+1, m+p]. \end{cases}$$

取β=2，r_i是违反约束程度的函数，即r_i是$G_i(\mathbf{x})$的一个非减函数，根据违反约束的量，惩罚因子r_i可在一个离散值集合中取值：

$$r_i = \begin{cases} R_{i1}, & 如果\ G_i(\boldsymbol{x}) \in (0, T_{i1}], \\ R_{i2}, & 如果\ G_i(\boldsymbol{x}) \in (T_{i1}, T_{i2}], \\ \vdots & \\ R_{iq}, & 如果\ G_i(\boldsymbol{x}) \in (T_{i,q-1}, \infty), \end{cases}$$

q是用户指定的约束水平的个数，R_{ij}是用户定义的权重，T_{ij}是用户定义的约束阈值。

处理约束的常用方法

- 动态惩罚方法
  - 将等式约束转换为不等式约束：

$$\min_{\boldsymbol{x}} \phi(\boldsymbol{x}),\ 这里\ \phi(\boldsymbol{x}) = f(\boldsymbol{x}) + \sum_{i=1}^{m+p} r_i G_i(x)$$

$$G_i(\boldsymbol{x}) = \begin{cases} (\max\{0, g_i(\boldsymbol{x})\})^{\beta}, & i \in [1, m], \\ (\max\{0, |h_i(\boldsymbol{x})| - \epsilon\})^{\beta}, & i \in [m+1, m+p]. \end{cases}$$

取$r_i=(ct)^{\alpha}$，这里c和α为常数，t为代数：

$$\phi(\boldsymbol{x}) = f(\boldsymbol{x}) + (ct)^{\alpha} M(\boldsymbol{x}),$$

$$M(\boldsymbol{x}) = \sum_{i=1}^{m+p} G_i(\boldsymbol{x}).$$

通常c取1/2, α取1或2

约束的惩罚随着代数增加。

处理约束的常用方法

- 自适应惩罚方法
  - 利用种群的反馈调整惩罚权重，将权重设为：

$$r_i(t+1)=\begin{cases} r_i(t)/\beta_1, & \text{如果是情况1,} \\ \beta_2 r_i(t), & \text{如果是情况2,} \\ r_i(t), & \text{其他,} \end{cases}$$

其中t为代数，β_1和β_2是满足$\beta_1>\beta_2>1$的常数。情况1意味着最好的个体在过去k代的每一代都是可行的，情况2意味着在过去的k代都没有可行个体；

  - 如果种群中最好的个体可行，减小约束权重就会容许种群中有更多不可行个体；如果种群中没有可行个体，增大约束权重有利于获得一些可行个体。

处理约束的常用方法

- 随机排名
  - 在约束进化算法中加入随机成分：有时根据适应度值对候选解排名，有时根据违反约束的程度对候选解排名（对个体的决策是随机的）
  - 在比较两个个体$\mathbf{x}_1$和$\mathbf{x}_2$时，认为$\mathbf{x}_1$优于$\mathbf{x}_2$，如果：

1）这两个解都可行且$f(\mathbf{x}_1)<f(\mathbf{x}_2)$；或

2）随机生成的数$r\sim U[0,1]$小于用户给定的概率P_f，并且$f(\mathbf{x}_1)<f(\mathbf{x}_2)$；或

3）上面的两个条件都不满足，但是$\mathbf{x}_1$比$\mathbf{x}_2$违反的约束少。

在对种群中所有个体进行比较和排序后，为生成下一代选择父代并进行交叉。对于大部分问题，$P_f\in(0.4,0.5)$会比较好。

处理约束的常用方法

- 单目标/多目标进化优化中的约束处理
  - 比较两个个体x_1和x_2的优劣

    1）x_1和x_2均为可行解；

    2）x_1可行，x_2不可行 或 x_2可行，x_1不可行 ；

    3）x_1和x_2均为不可行解。

    单目标进化优化：

    1）选择适应度值更好的个体；

    2）选择可行个体；

    3）选择约束违背少的个体

处理约束的常用方法

- 单目标/多目标进化优化中的约束处理

多目标进化优化：

重新定义支配，如果$\mathbf{x}_1$支配$\mathbf{x}_2$，需满足以下任一条件：

1） $\mathbf{x}_1$可行，$\mathbf{x}_2$不可行；

2） $\mathbf{x}_1$和$\mathbf{x}_2$均不可行，但$\mathbf{x}_1$较少地违背约束；

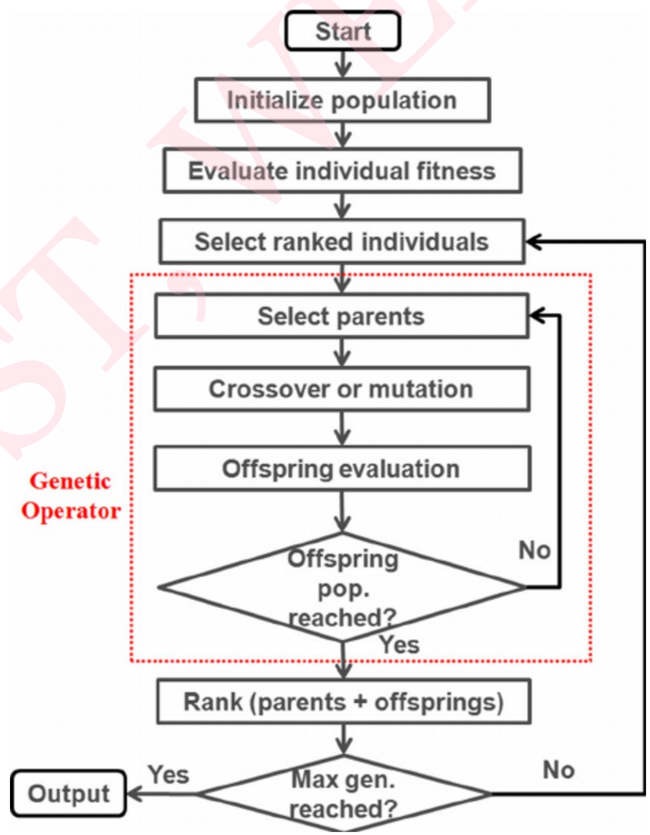
3） $\mathbf{x}_1$和$\mathbf{x}_2$均可行，$\mathbf{x}_1$支配$\mathbf{x}_2$

处理约束的常用方法

- 单目标/多目标进化优化中的约束处理

多目标进化优化：

Constrained NSGA-II：



结束