智能优化算法

NSGA-II

(Non-dominated Sorting Genetic Algorithm)

0 前置知识

- 智能优化算法基本原理和流程
  - 【通俗易懂讲算法-最优化之遗传算法（GA）】
    https://www.bilibili.com/video/BV14a411C7s8
  - 【通俗易懂讲算法-最优化之粒子群优化（PSO）】
    https://www.bilibili.com/video/BV1uY41187rK

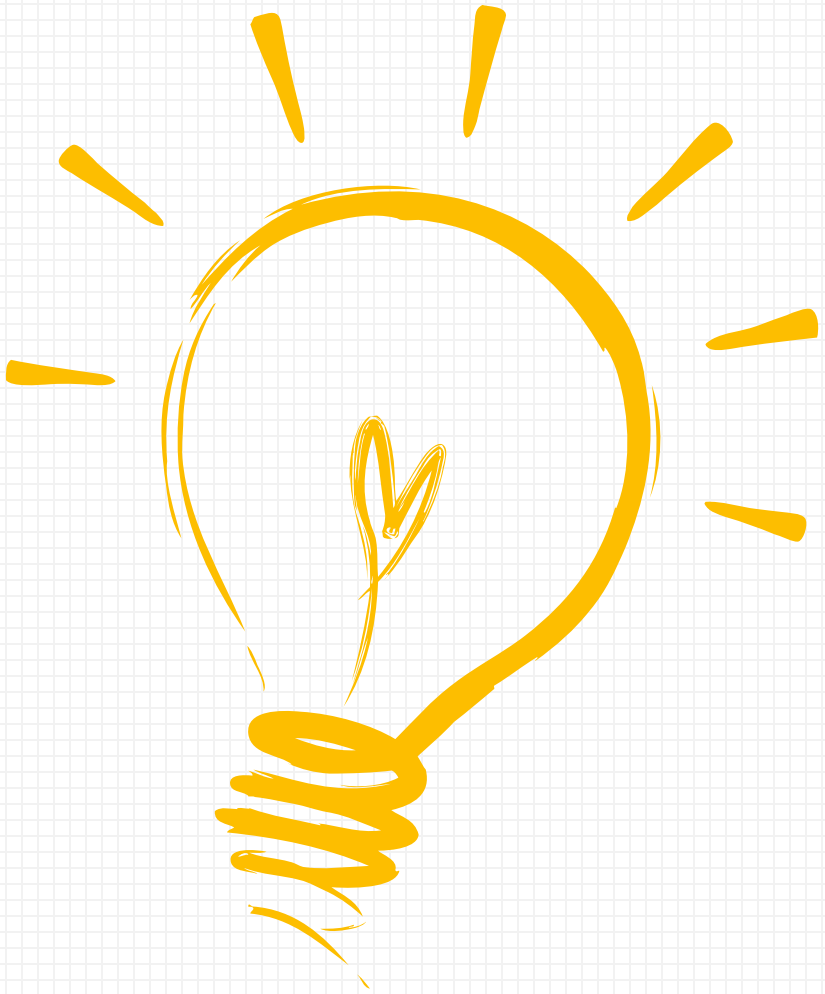
基本原理

这些算法本质一样, 会一个其他就都会了

- 基本编程知识
  - Python
  - MATLAB(本次视频课用到)

拓展知识

CONTENTS



di yi zhang jie

第一章节

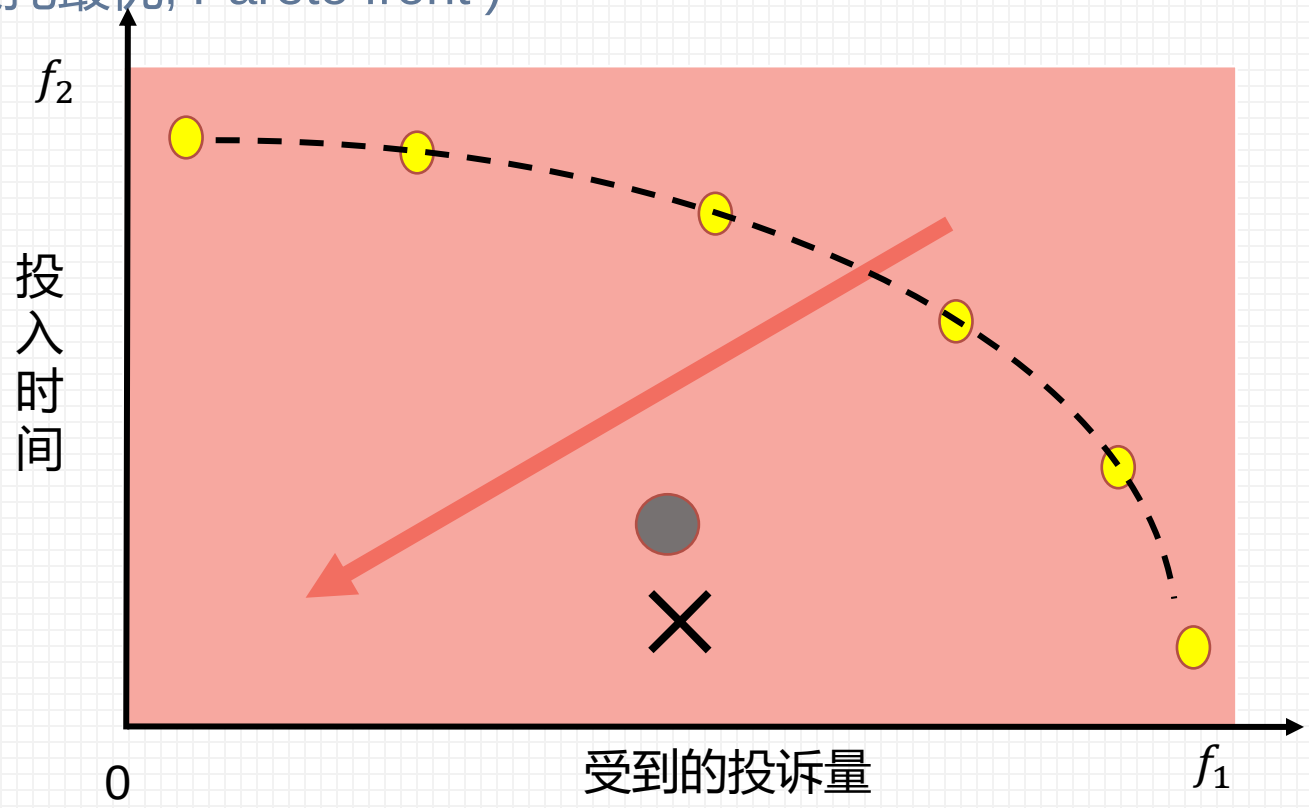
算法背景

1.1 背景和概念

多目标优化是涉及**多个目标函数同时优化**的数学问题.

需要在两个或多个相互冲突的目标之间进行权衡的情况下作出**最优决策**.

帕累托前沿(帕累托最优, Pareto front)



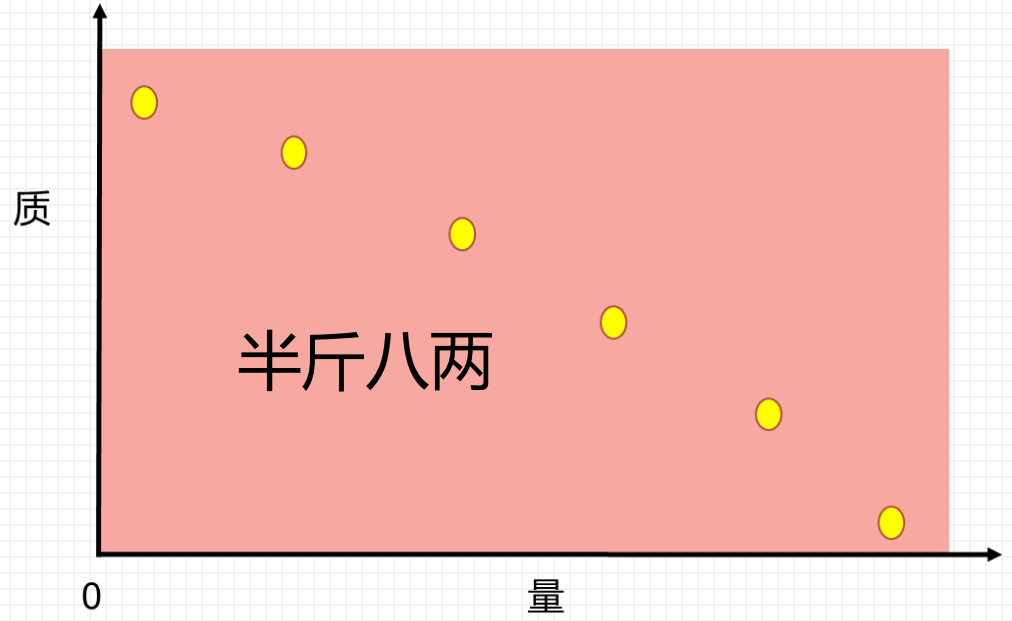
di yi zhang jie

第二章节

算法原理

2.1 基本原理

- 智能优化基本流程
  - 初始化N个解(个体)
  - 计算函数值(或者适应度)
  - 利用旧解产生新解
    - 各种策略: GA, PSO , 灰狼优化等 (本质一样)
  - 选择得到新一轮的解
    - 单目标: 谁的y值小, 谁就好(最小化问题)
      - 旧的解50个, 新的解50个 , 直接整体排序, 保留50个即可
    - 多目标:
      - 旧的解50个,新的解50 , 怎么排序? 谁更好?



多目标优化真正的核心难点: 怎么评价解得好坏, 即怎么排序

2.1 基本原理 - 非支配排序

支配(Dominate)

- Solution **X** is said to **dominate** solution **Y** if and only if:

1. Solution X is **no worse than solution Y in all objectives functions** and
2. Solution X is **better than solution Y in at least one objective function.**

	X	Y
Max Obj1	4	4
Max Obj2	0.3	0.2

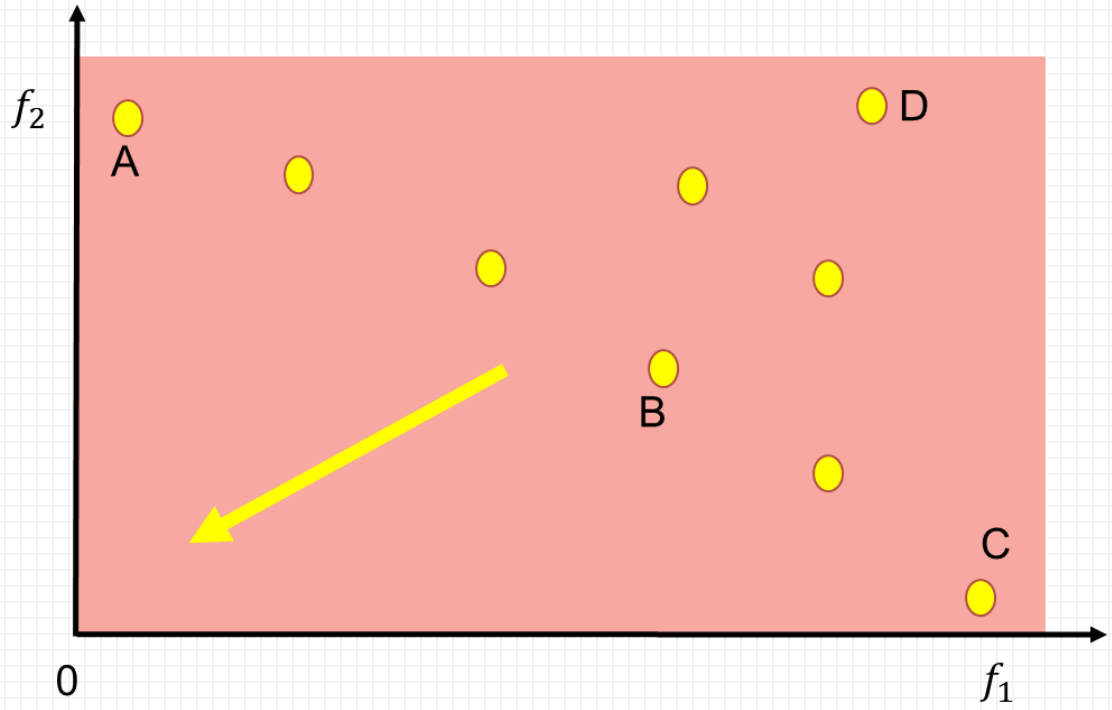
Does solution X dominates solution Y?

YES.

	X	Y
Max Obj1	5	4
Max Obj2	0.1	0.25

Does solution X dominates solution Y?

NO.



- $B \prec D$、$A \prec D$ (以右图$minimize$为例)
- A与B, 互相无法支配 (看成一样的)
- 代码实现 : flag = all(A<=B) && any(A<B);

$A \prec D$ 其实想说, A 严格比 D 好

2.1 基本原理 - 非支配排序

拥挤距离(Crowding - Distance)

crowding-distance-assignment($\mathcal{I}$)

$l = |\mathcal{I}|$ — number of solutions in $\mathcal{I}$

for each i, set $\mathcal{I}[i]_{\text{distance}} = 0$ — initialize distance

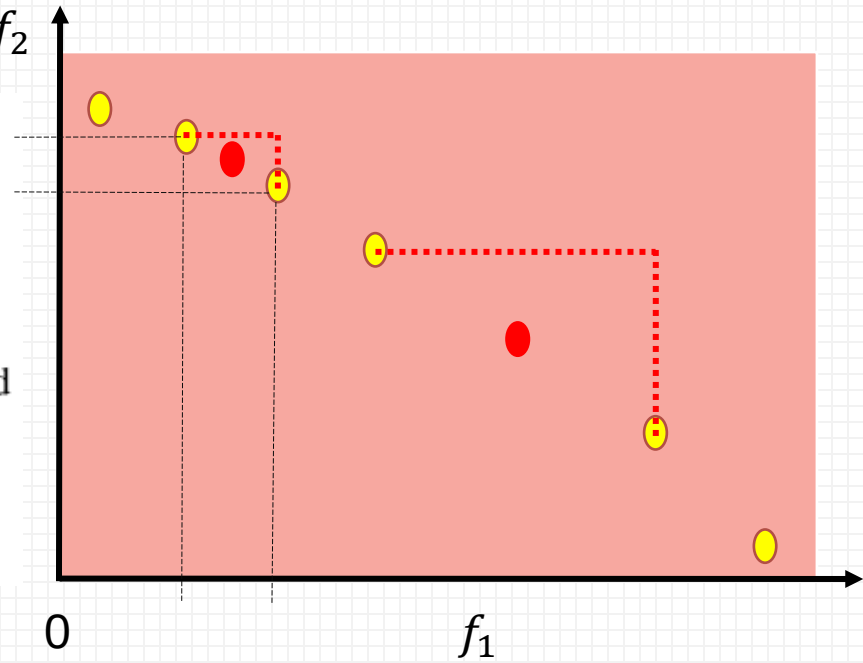
for each objective m

$\mathcal{I} = \text{sort}(\mathcal{I}, m)$ — sort using each objective value

$\mathcal{I}[1]_{\text{distance}} = \mathcal{I}[l]_{\text{distance}} = \infty$ — so that boundary points are always selected

for $i = 2$ to $(l - 1)$ — for all other points

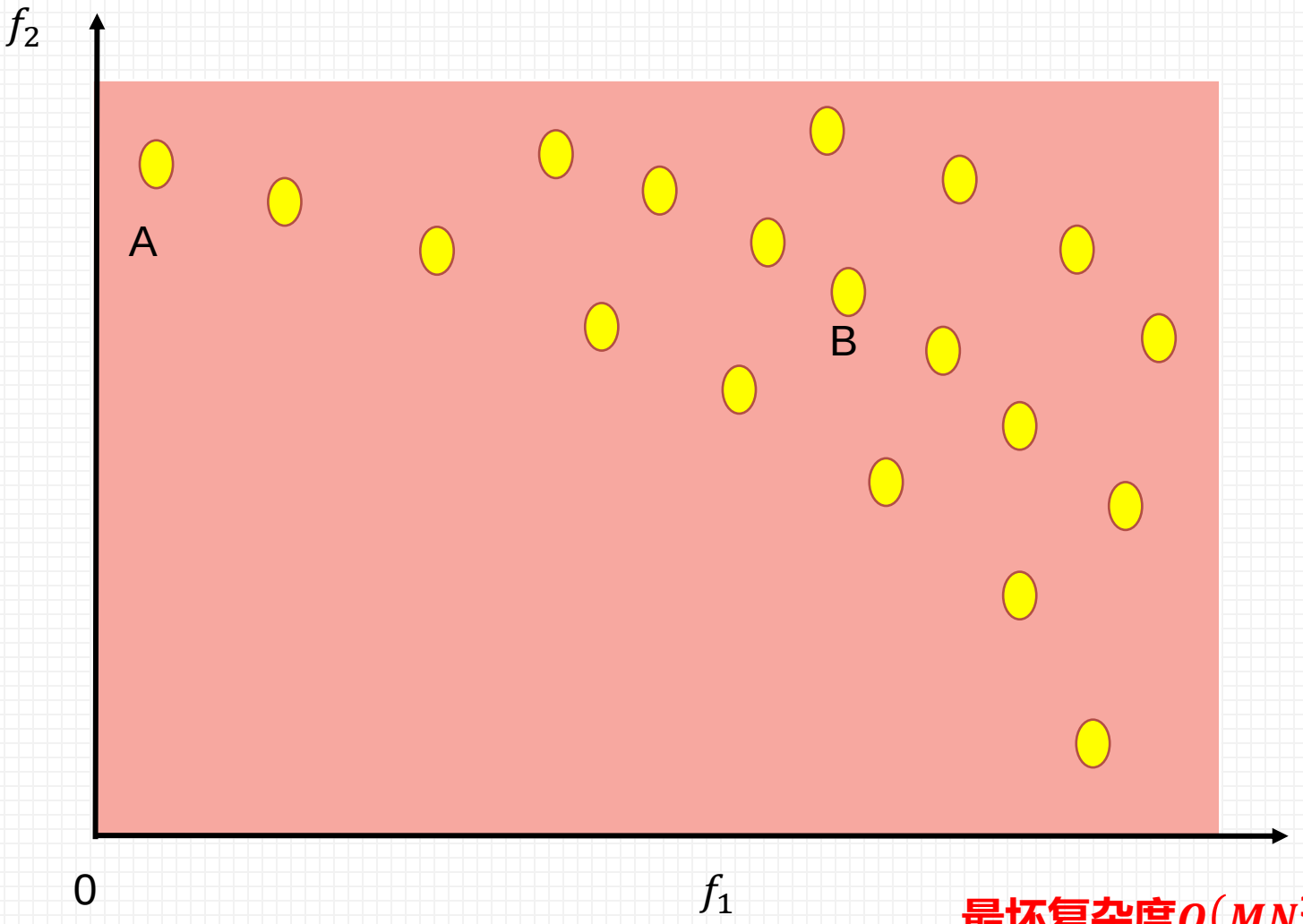
$\mathcal{I}[i]_{\text{distance}} = \mathcal{I}[i]_{\text{distance}} + (\mathcal{I}[i+1].m - \mathcal{I}[i-1].m)/(f_m^{\max} - f_m^{\min})$



- 拥挤距离越大:
  - 这里解比较空旷
  - 选择拥挤距离大的解, 有利于种群多样性 (可能)
- 因此, 非支配解之间, 也能进行比较大小 .
- **比较大小的问题 , 得到了解决**

2.1 基本原理 - 非支配排序

非支配排序(Non-dominated Sorting)

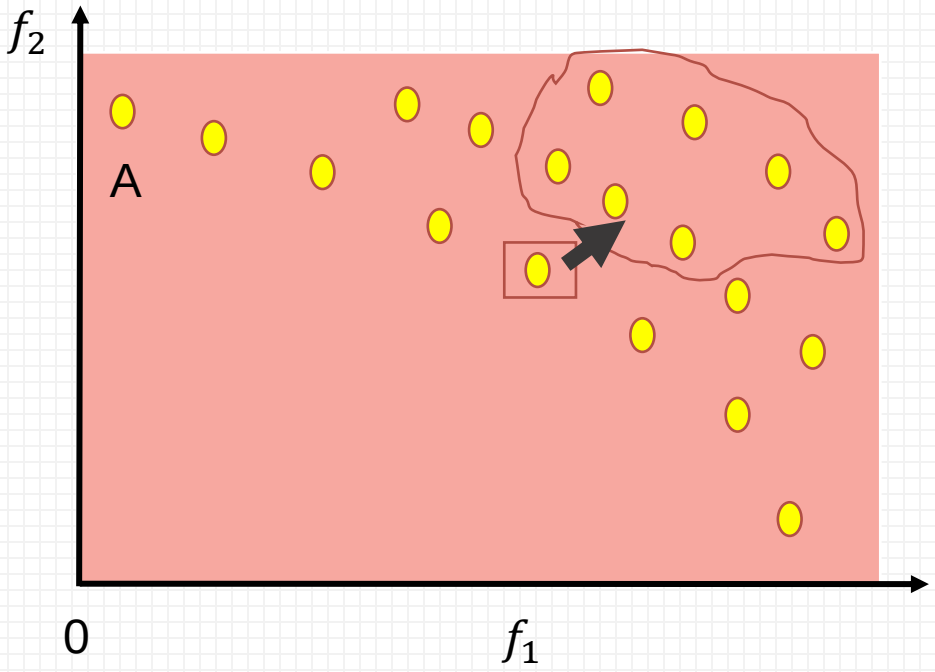



```
Pop = pop
While Pop :
    for pi in Pop：
        for pj in Pop:
            if pj ≺ pi： pi.dominatecount += 1
    S = find(pi.dominatecount == 0)
    S .Rank = i # 标志这是第几层
    F.append(S) # 记录每一层的个体
    Pop = Pop - S # 下一轮两两比较
    i += 1
```

最坏复杂度$O(MN^3)$, M是obj个数,N是pop size

2.1 基本原理 - 非支配排序

NSGA - II 快速非支配排序



$S_p = \{\dots\}$,当前个体p所支配的那些个体

$n_p = count$,能够支配当前个体p的个数

最坏复杂度$O(MN^2)$

M是obj个数,*N* 是*pop size*

```
fast-non-dominated-sort(P)
for each p ∈ P
   S_p = ∅
   n_p = 0
   for each q ∈ P
      if (p ≺ q) then                    If p dominates q
         S_p = S_p ∪ {q}                 Add q to the set of solutions dominated by p
      else if (q ≺ p) then
         n_p = n_p + 1                   Increment the domination counter of p
   if n_p = 0 then                       p belongs to the first front
      p_rank = 1
      F_1 = F_1 ∪ {p}
i = 1                                    Initialize the front counter
while F_i ≠ ∅
   Q = ∅                                 Used to store the members of the next front
   for each p ∈ F_i
      for each q ∈ S_p
         n_q = n_q − 1
         if n_q = 0 then                 q belongs to the next front
            q_rank = i + 1
            Q = Q ∪ {q}
   i = i + 1
   F_i = Q
```

2.1 基本原理

- 多目标优化基本流程
  - 初始化N个解(个体)
  - 计算函数值(或者适应度)
  - 利用旧解产生新解
    - 各种策略: GA, PSO , 灰狼优化等 (本质一样), **本文是GA**
  - 通过比大小选择得到新一轮的解
    - 非支配排序, 确定Rank
    - 对于同一个Rank, 拥挤距离越大越好
  - 循环

怎么评价解得好坏, 即怎么排序：解决

di san zhang jie

第三章节

算法分析

3.1 算法分析

- 多目标优化基本流程
  - 初始化N个解(个体)
    - **如何初始化，什么样的初始化对求解有帮助?**
  - 计算函数值(或者适应度)
  - 利用旧解产生新解
    - 各种策略: GA, PSO，灰狼优化等 (本质一样)
      - **对策略改进，GA：交叉变异, PSO：速度，系数 ，算法融合等等**
    - 通过比大小选择得到新一轮的解
      - 非支配排序, 确定Rank
        - **复杂度？$O(MN^2)$ -> 更小**
      - 对于同一个Rank, 拥挤距离越大越好
        - **解得多样性, 其他方式？基于熵的方式? 局部密度?**

di si zhang jie

第四章节

算法拓展

4.1 算法拓展

略

di wu zhang jie

第五章节

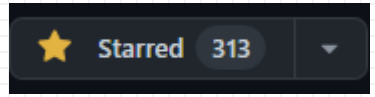
案例实操

5.1 案例实操

- 代码实现：
  - MATLAB
    - https://yarpiz.com/56/ypea120-nsga2
  - Python
    - 直接搜~ NSGA II code
- 测试函数
  - https://en.wikipedia.org/wiki/Test_functions_for_optimization
  - https://www.sfu.ca/~ssurjano/optimization.html
  - https://machinelearningmastery.com/2d-test-functions-for-function-optimization/
  - https://xloptimizer.com/projects/toy-problems/mop2-function-multi-objective

5.2 广告时间

PPT和代码(Github,评论区)：https://github.com/CHENHUI-X/My-lecture-slides-and-code



点赞支持~ 一键三连 ~

https://space.bilibili.com/294132471

参　考　资　料

[1] K. Deb, A. Pratap, S. Agarwal and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," in IEEE Transactions on Evolutionary Computation, vol. 6, no. 2, pp. 182-197, April 2002, doi: 10.1109/4235.996017.

THANKS

谢谢观看