Machine Learning

CHAPTER 9 Genetic Algorithms

9.1 动机

- 模拟生物进化
  - 不断变异、重组当前最好假设生成后续的假设

- GA 特点:
  - 进化具有很好的鲁棒性
  - 假设空间中，假设的各个部分相互作用，每个假设对适应度的影响难以建模
  - 易于并行计算

9.2 Genetic Algorithms

- 假设适应度（Hypothesis fitness）
  - Fitness
  - Best hypothesis has the highest fitness

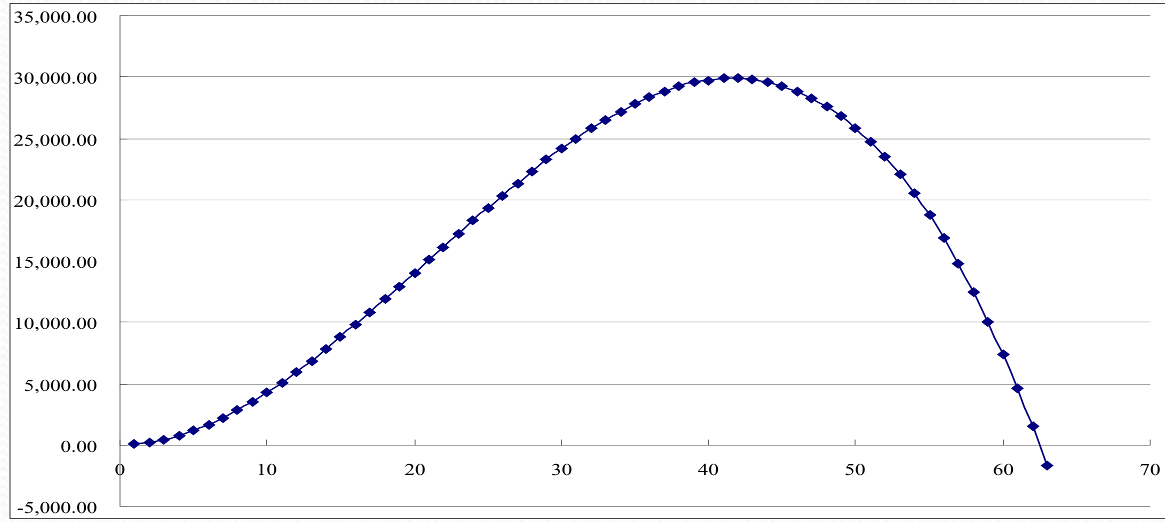
- 算法结构:
  - 迭代更新种群（ population）
  - 对种群的每个成员进行评估
  - 产生新的种群

Let's start from a problem

$$Max \;\; F(X) = -0.83X^3 + 52X^2 - 8.5X + 8$$

$$Subject\ to: \;\; 0 \le X \le 61$$

$$X \in Integer$$



- □ □ □ ?
  - □ □ □ , □ □ □ □ □ □ □ □ □
  - □ □ □ □ (□ □ , □ □ □ □)□
  - □ □ , □ □ □ □ □ □
  - □ □ ?

A dumb solution

A "blind generate and test" algorithm:

Repeat

    Generate a random possible solution

    Test the solution and see how good it is

Until solution is good enough

Can we use this dumb idea?

- Sometimes - yes:
  - if there are only a few possible solutions
  - and you have enough time
- For most problems - no:
  - many possible solutions
  - with no time to try them all

A “less-dumb” idea (GA)

产生一组随机解

Repeat

  评估集合中的每一个解(rank them)

  删除集合中的不良解

  复制部分优良解

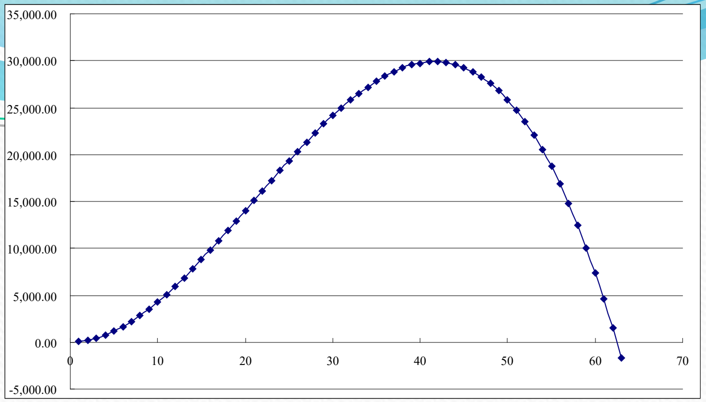
  对部分解进行小的改变

Until best solution is good enough

$$Max \;\; F(X) = -0.83X^3 + 52X^2 - 8.5X + 8$$

$$Subject\ to:\ \ 0 \le X \le 61$$

$$X \in Integer$$



Step 1:Encoding（编码）

							X	***F(X)***
pop 1	1	0	1	0	1	1	***43***	***29799.69***
pop 2	0	0	0	1	1	0	***6***	***1649.72***
pop 3	1	0	0	0	1	1	***35***	***27824.25***

Step 2: Generate population（产生群体）

pop 1	**0**	**1**	**1**	**1**	**0**	**1**	*29*
pop 2	**1**	**0**	**1**	**0**	**1**	**0**	*42*
pop 3	**0**	**1**	**1**	**1**	**1**	**1**	*31*
pop 4	**0**	**1**	**0**	**1**	**0**	**1**	*21*
pop 5	**1**	**0**	**0**	**1**	**1**	**1**	*39*
…	.	.	.	.	.	.	.
…	.	.	.	.	.	.	.
pop 19	**0**	**1**	**0**	**0**	**0**	**1**	*17*
pop 20	**0**	**0**	**1**	**1**	**1**	**1**	*15*

Step2 Selection（选择）

							X	F(X)	survival rate
Pop 1	0	1	1	1	0	1	29	23250.63	5.77%
Pop 2	1	0	1	0	1	0	42	29885.96	7.41%
Pop 3	0	1	1	1	1	1	31	24989.97	6.20%
Pop 4	0	1	0	1	0	1	21	15074.87	3.74%
Pop 5	1	0	0	1	1	1	39	29533.73	7.33%
Pop 6	1	0	1	1	1	1	47	28303.41	7.02%
Pop 7	0	1	0	1	0	1	21	15074.87	3.74%
Pop 8	0	1	0	0	0	1	17	10813.71	2.68%
Pop 9	0	0	1	1	1	1	15	8779.25	2.18%
Pop 10	1	1	0	1	1	1	55	18749.25	4.65%
Pop 11	0	1	1	1	0	1	29	23250.63	5.77%
Pop 12	0	1	0	1	1	1	23	17221.89	4.27%
Pop 13	1	0	1	0	1	1	43	29799.69	7.39%
Pop 14	0	1	1	0	1	0	26	20350.92	5.05%
Pop 15	1	0	0	0	1	1	35	27824.25	6.90%
Pop 16	0	0	1	1	1	1	15	8779.25	2.18%
Pop 17	0	1	0	1	1	1	23	17221.89	4.27%
Pop 18	0	0	0	1	0	1	5	1161.75	0.29%
Pop 19	1	0	1	0	1	1	43	29799.69	7.39%
Pop 20	0	1	1	1	0	1	29	23250.63	5.77%

Survival Rate= fitness/ sum(fitness)

Step 3: Crossover（交叉）

pop 1	0	1	1	1	0	1	*29*
pop 2	1	0	1	0	1	0	*42*
pop 3	0	1	1	1	1	1	*31*
pop 4	0	1	0	1	0	1	*21*
pop 5	1	0	0	1	1	1	*39*
…	.	.	.	.	.	.	.
…	.	.	.	.	.	.	.
pop 19	0	1	0	0	0	1	*17*
pop 20	0	0	1	1	1	1	*15*

Randomly select parents

Parent 1	0	1	1	1	0	1
Parent 2	1	0	0	1	1	1

Crossover point

							X	***F*(*X*)**
Offspring 1	0	1	1	1	1	1	*31*	*24989.97*
Offspring 2	1	0	0	1	0	1	*37*	*28839.51*

Step 4: Mutation（变异）

	Chromosome	*X*	*F*(*X*)
Offspring 1	0 1 1 1 1 1	31	24989.97
Offspring 2	1 0 0 1 0 1	37	28839.51
Offspring 3	0 0 1 0 1 0	10	4293.00
……			
Offspring 20	0 0 0 1 1 0	6	1649.72

Offspring 2 →

Chromosome	*X*	*F*(*X*)
1 0 0 1 1 1	39	29533.73

Randomly select mutation point

Step 5: Next round-back to Step2

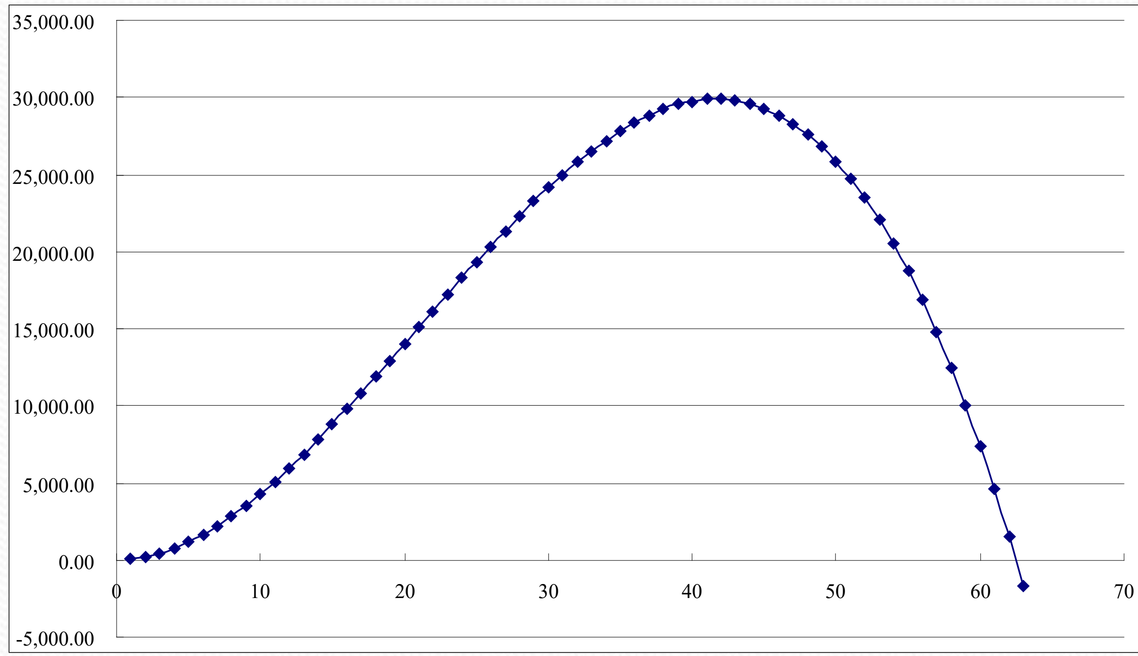
When converged!

$$Max\ F(X) = -0.83X^3 + 52X^2 - 8.5X + 8$$

$$Subject\ to:\ 0 \le X \le 61$$

$$X = Integer$$

							X	$F(X)$
Pop 1	1	0	1	0	1	0	42	29885.96
Pop 2	1	0	1	0	1	0	42	29885.96
Pop 3	1	0	1	0	1	0	42	29885.96
Pop 4	1	0	1	0	1	0	42	29885.96
Pop 5	1	0	1	0	1	0	42	29885.96
Pop 6	1	0	1	0	1	0	42	29885.96
Pop 7	1	0	1	0	1	0	42	29885.96
Pop 8	1	0	1	0	1	0	42	29885.96
Pop 9	1	0	1	0	1	0	42	29885.96
Pop 10	1	0	1	0	1	0	42	29885.96
Pop 11	1	0	1	0	1	0	42	29885.96
Pop 12	1	0	1	0	1	0	42	29885.96
Pop 13	1	0	1	0	1	0	42	29885.96
Pop 14	1	0	1	0	1	0	42	29885.96
Pop 15	1	0	1	0	1	0	42	29885.96
Pop 16	1	0	1	0	1	0	42	29885.96
Pop 17	1	0	1	0	1	0	42	29885.96
Pop 18	1	0	1	0	1	0	42	29885.96
Pop 19	1	0	1	0	1	0	42	29885.96
Pop 20	1	0	1	0	1	0	42	29885.96



A prototypical genetic algorithm

- GA(Fitness, fitness_threshold, p,r,m)

Fitness: 适应度函数;

fitness_threshold:终止判据的阈值

p:群体中包含的假设数量;

r:通过交叉替换成员的比例;

m:变异率

initialize population : P

Evaluate : For each h in P, compute Fitness(h)

while[max Fitness(h)] < Fitness_threshold do

{

create a new generation, P_S:

1.Select: select (1-r)p members of P to add to P_S.

$$Pr(h_i)=Fitness(h_i) \;/\; \sum Fitness(h_j)$$

2.Crossover: select r*p/2 pairs of hypotheses from P.

applying the crossover operator. Add all offspring to P_S

3. Mutate: choose m percent of the members of P_S.

invert one randomly selected bit in its representation.

4. update: $P=P_S$

5.Evaluate: for each h in P, compute Fitness(h)

}

Return the hypothesis from P that has the highest fitness

Parameters of GA

- GA has following parameters:
  - Crossover rate（交叉率）
  - Mutation rate（变异率）
  - Population size（群体大小）
  - Encoding（编码）
  - Crossover and mutation type（交叉、变异类型）
  - Selection（选择）
- 参数如何确定，无确定方法.
  - 根据具体问题, 实验确定

9.2.1 Representing Hypotheses

- 假设表示为位串
  - 易于被遗传算子操作

- Example: 编码 if-then 规则
  - If wind=strong, then playTennis =yes
  - Attribute coding（属性编码）
    - Wind：Strong，weak
    - 两位长位串：**10**
    - 规则中未被约束的属性编码
      - Outlook=111

9.2.1 Representing Hypotheses

- rule coding
  - concatenating the corresponding bit strings
  - 111 10 10

- Representation of sets of rules：
  - concatenating bit strings of individual rules

- 语法正确的位串应能表示有意义的假设
  - 111 10 11?

9.2.2 Genetic operators

- The most common operators : crossover and mutation
- The crossover（交叉） operator：
  - Produces two new offspring from two parents strings by copying selected bits from each parent
  - crossover mask:
    - Single point crossover

A□ 1011011100　1111111100　A'□ 1011011111

B□ 0001110011　　　　　　　B'□ 0001110000

- Two-point crossover

A: 1011011100　0001111100　A'□ 0001011111

B□ 0001110011　　　　　　　B'□ 1011110000

- Uniform crossover

A: 1011001100　0001101100　A'□ 0001011111

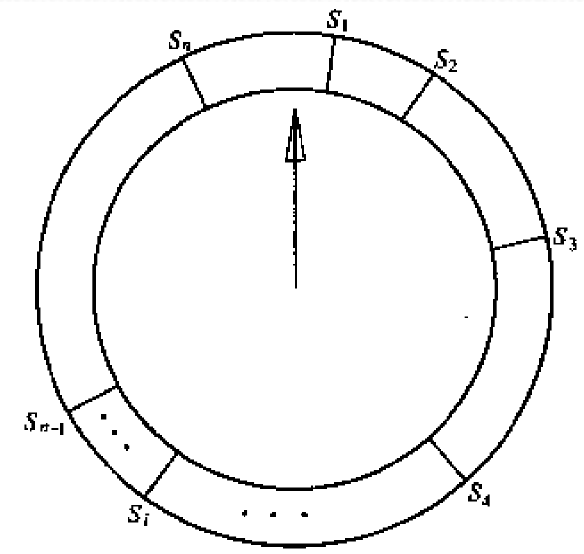
B□ 0001110011　　　　　　　B'□ 1011100000

9.2.2 Genetic operators （2）

- Mutation operator：
  - Produces offspring from a single parent and produces small changes to the bit string
  - Mutation is often performed after crossover has been applied

9.2.3 Fitness Function and Selection

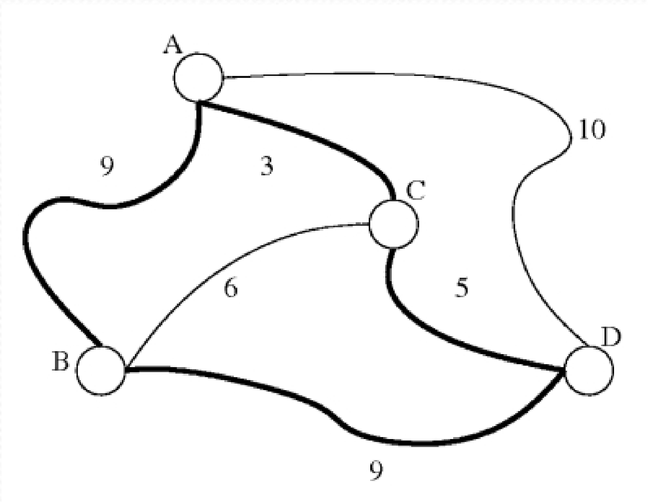
- 适应度函数定义了候选假设的排序准则
- 选择假设的概率方法
  - 适应度比例选择(roulette wheel selection)
  - 锦标赛选择
    - Randomly select two h
    - p,(1-p) -> h_{high}, h_{low}
    - A more diverse population
  - 排序选择
    - Sorted by fitness
    - Probability is proportional to its rank



TSP

- **The decision version of this problem has been proven to be NP-Complete!**
- **Existing solutions:**
  - **heuristics,**
  - **cutting-plane methods,**
  - **branch-and-cut**
- **How to solve it by GA?**

(1) Encoding



	1^{st} city on tour	*2^{nd} city on tour*		*last city on tour*
City #				

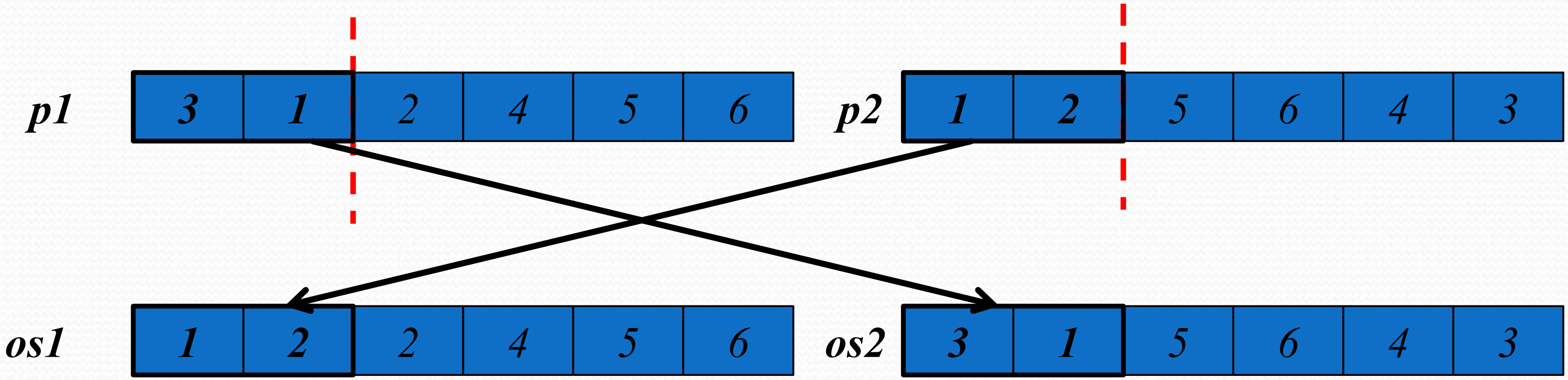
- For example, we have 6 cities, and a tour is 3-1-2-4-5-6

3	***1***	*2*	*4*	*5*	*6*

(2) Fitness Evaluation

- Given a chromosome, simply calculate the total distance.
- The shorter the distance, the higher the fitness of the chromosome

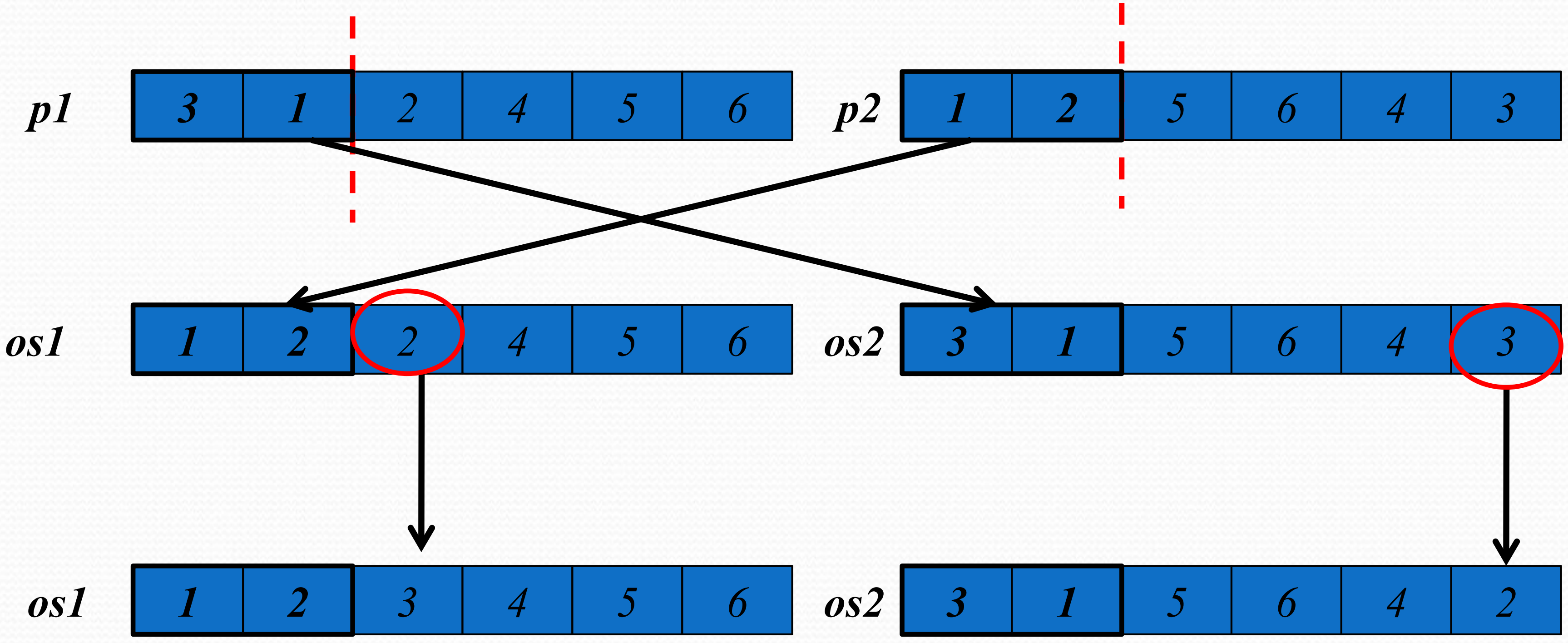
(3) Cross-over



- Do you find any problem?

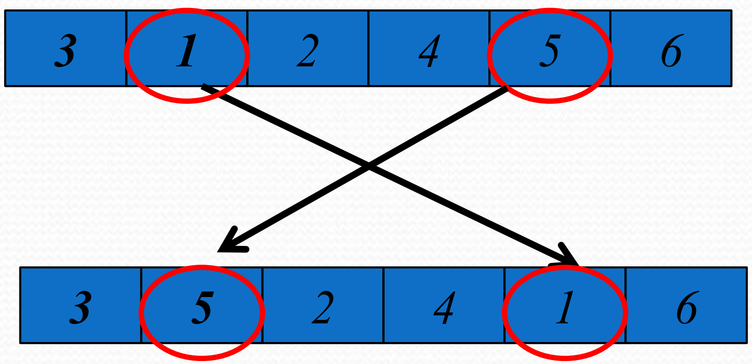
How to solve this problem?

- Repair operator

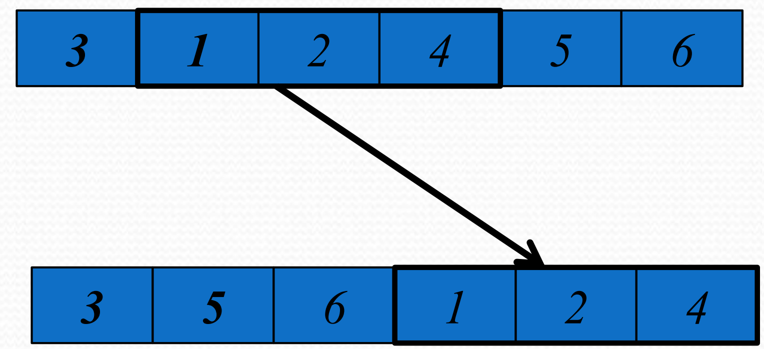


(3) Mutation

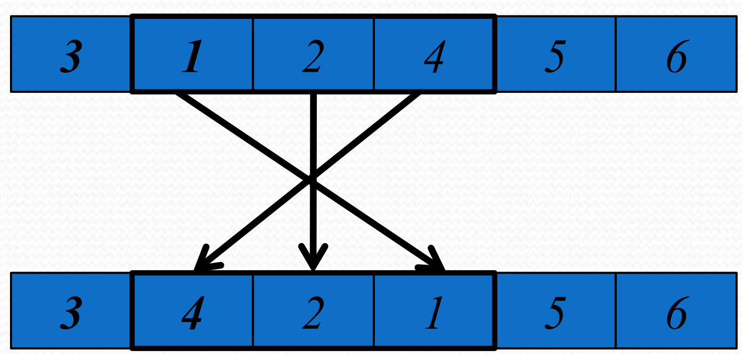
- Exchange mutation



- Displacement mutation



- Simple inversion mutation



9.4 Hypothesis Space Search

- GAs 使用随机柱状搜索寻找最大适应度假设
  - The gradient descent search moves smoothly from one h to a new h
  - GA搜索的移动可能非常突然
    - less likely to fall into the local minima
- 实际应用困难: 拥挤问题（ crowding）
  - 高适应度值个体迅速繁殖，群体包含过多相似个体
  - 降低种群多样性，减慢进化过程

Hypothesis space search（2）

- 避免拥挤的策略
  - 修改选择函数
    - 使用锦标赛或排序选择
  - 适应度共享
    - 根据相似性个体的数量，降低个体的适应度
  - 限制可重组生成后代的个体种类
    - 只允许最相似的个体进行重组
    - 按空间分配个体，仅允许相邻个体重组

Summary

- 遗传算法优点:
  1. 无需先验知识或计算误差函数的梯度信息
  2. 可设计多目标函数
  3. 并行性
  4. 可很容易地用于对庞大、难以理解的假设空间进行搜索
  5. 可广泛应用于优化问题的求解

Summary

遗传算法的缺点:

1. 问题的表示
2. 适应度函数的定义
3. 过早收敛
4. 参数的选择问题：如种群大小，变异率，交叉率，选择的策略等
5. 无法应用梯度信息
6. 终止条件难判断
7. 频繁计算适应度，计算量较大