现代智能优化算法

- 授课教师：张玉洁
- 院　　系：数理学院

目录

现代优化算法

禁忌搜索算法
模拟退火算法
遗传算法
人工神经网络
蚁群算法
粒子群算法
混合算法

特点：

- 基于客观世界中的一些自然现象；
- 建立在计算机迭代计算的基础上；
- 具有普适性，可解决实际应用问题。

如何解决问题

考虑一个一元方程： $ax^2+bx+c=0$

经典数学理论的解法：

Viete 定理： $x=\dfrac{-b\pm\sqrt{b^2-4ac}}{2a}$

由公理、定理形成的演绎逻辑体系

优点：准确、快速、有明确数学意义

不足：只能解决有限问题、解法是特定

的无法解决高次问题及更复杂问题。

如何解决问题

数值解法：

通过引入一些假设来获得问题的近似解

$$x^{10}+e^{x}=100\,(x>0)$$

$$\longrightarrow\quad x=\sqrt[10]{100-e^{x}}\qquad(1)$$

取初始值为1，代入得：

1　1.58053　1.57702　1.57705

优点：可求解问题的范围比较广泛

不足：解法的特殊性，要求迭代收敛

传统算法的局限

- 解法是特定的

  ---- 不同的问题需要使用不同的解法

- 所能解决的问题是有限的

  ---- 和数学工具直接相关，若没有解决问题的数学方法，则难以解决

最优化理论

最优化理论的三大经典算法：

模拟退火法(SA)、

神经网络(NN)、

遗传算法(GA)

近几年的赛题越来越复杂，很多问题没有什么很好的模型可以借鉴，于是这三类算法很多时候可以派上用场。

最优化问题(Optimization Problem)

最优化问题:

$$Minimize \quad f(x) = f(x_1, x_2, \cdots, x_n)$$

$$subject\ to \quad x = (x_1, x_2, \cdots, x_n) \in S \subset X$$

组合优化问题(Combinatorial Optimization Problem) : 最优化问题中的解空间X或S由离散集合构成。其中很多问题是NP完全(Nondeterministic Polynomial Completeness)问题.

传统的优化问题

待解决的问题

连续性问题，以微积分为基础，*规模较小*

传统的优化方法

理论上的准确与完美，主要方法：线性与非线性规划、动态规划、多目标规划、整数规划等；排队论、库存论、对策论、决策论等。

传统的评价方法

算法收敛性、收敛速度

现代优化问题

现代优化算法又称智能优化算法或现代启发式算法，是一种具有全局优化性能、通用性强、且适合于并行处理的算法。这种算法一般具有严密的理论依据，而不是单纯凭借专家经验，理论上可以在一定的时间内找到最优解或近似最优解。

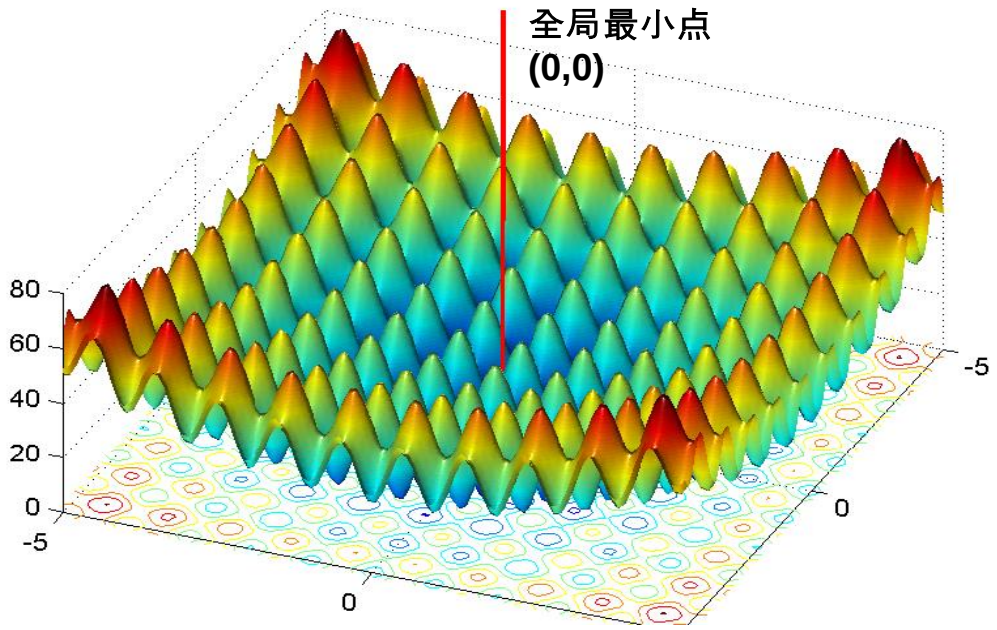
现代优化问题特点

它们的共同特点：都是从任一解出发，按照某种机制，以一定的概率在整个求解空间中探索最优解。由于它们可以把搜索空间扩展到整个问题空间，因而具有全局优化性能。

全局优化

$$Ras(x) = 20 + x_1^2 + x_2^2 - 10(\cos 2\pi x_1 + \cos 2\pi x_2)$$

Rastrigin's Function



现代优化算法

特点：

1)不依赖于初始条件；

2)不与求解空间有紧密关系,对解域无可微或连续的要求;容易实现，求解稳健。

3)但收敛速度慢,能获得全局最优;适合于求解空间不知的情况。

4)SA,GA可应用于大规模、多峰多态函数、含离散变量等全局优化问题;求解速度和质量远超过常规方法。

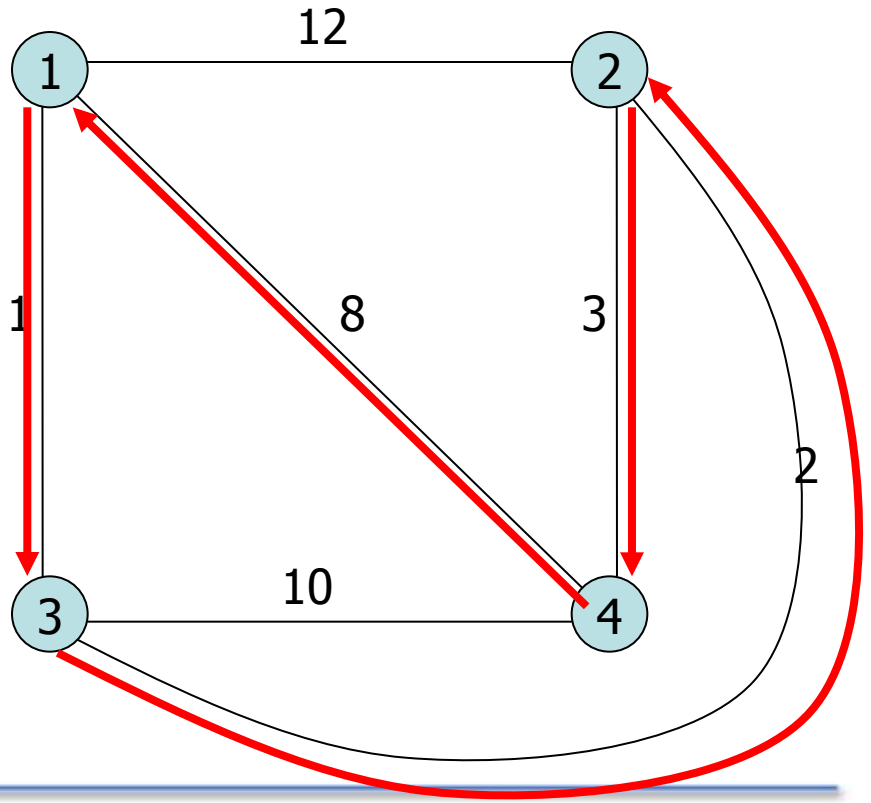
常用的现代优化算法

- ✓ 遗传算法　**Genetic Algorithm**
- ✓ 模拟退火算法　**Simulated Annealing**
- ✓ 禁忌搜索算法　**Tabu Search**
- ✓ 神经网络算法
- ✓ 群智能算法，包括蚁群算法(***Ant Colony Optimization***)、粒子群算法(***Particle Swarm Optimization***)
- ✓ 微分进化算法（**Differential Evolution**）

搜索示例：TSP问题

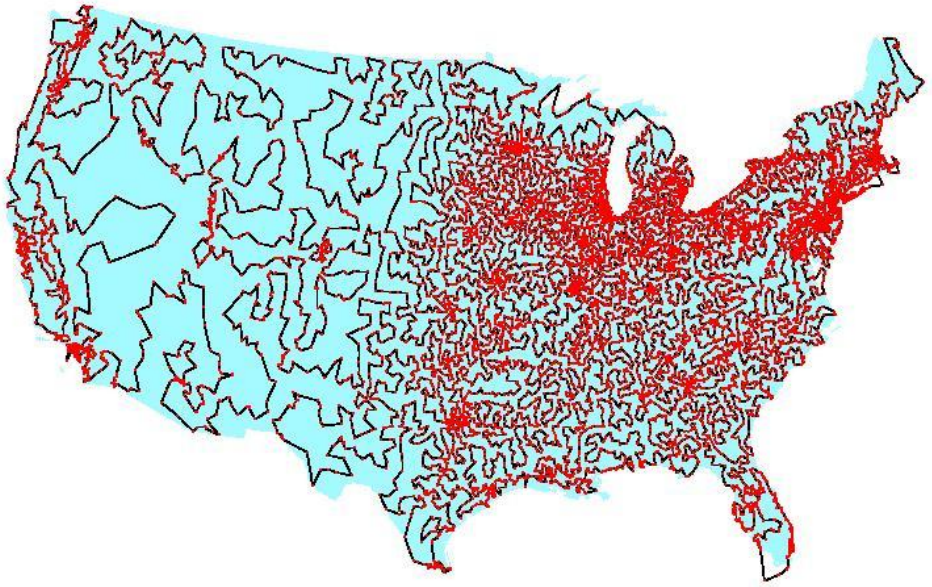
典型问题——旅行商问题（Traveling salesman problem, TSP）

给定*n*个城市和两两城市之间的距离，要求确定一条经过各城市当且仅当一次的最短路线。



TSP搜索的困难

计算复杂度：指数灾难



城市数	24	25	26	27	28	29	30	31
计算时间	1 sec	24 sec	10 min	4.3 hour	4.9 day	136.5 day	10.8 year	325 year

Part2　模拟退火法

模拟退火算法及模型

算法的提出

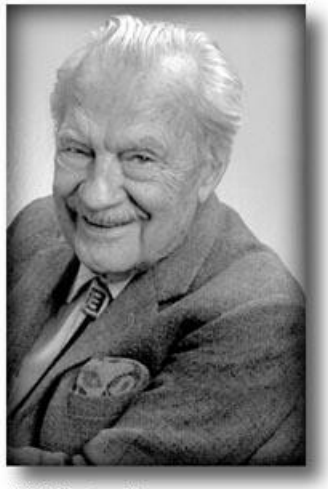
模拟退火算法最早的思想由Metropolis等（1953）提出，1983年Kirkpatrick等将其应用于组合优化。

算法的目的

解决NP复杂性问题；

克服优化过程陷入局部极小；

克服初值依赖性。



Nick Metropolis

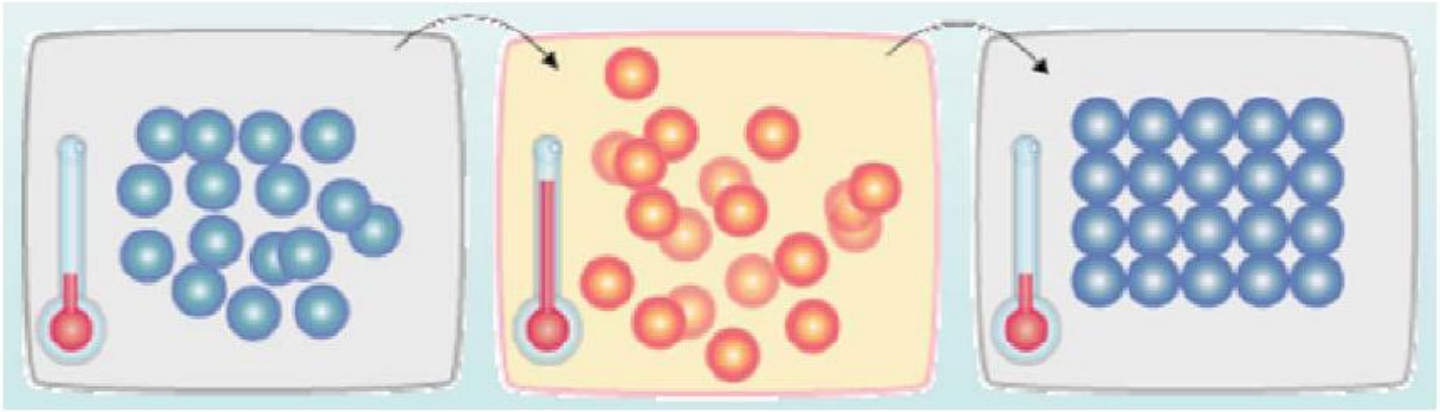
算法简介

模拟退火算法得益于材料统计力学的研究成果。统计力学表明材料中粒子的不同结构对应于粒子的不同能量水平。在高温条件下，粒子的能量较高，可以自由运动和重新排列。在低温条件下，粒子能量较低。如果从高温开始，非常缓慢地降温（这个过程被称为退火），粒子就可以在每个温度下达到热平衡。当系统完全被冷却时，最终形成处于低能状态的晶体。

物理退火过程

什么是退火：

退火是指将固体加热到足够高的温度，使分子呈随机排列状态，然后逐步降温使之冷却，最后分子以低能状态排列，固体达到某种稳定状态。



物理退火原理

物理退火过程

加温过程——增强粒子的热运动，消除系统原先可能存在的非均匀态；

等温过程——对于与环境换热而温度不变的封闭系统，系统状态的自发变化总是朝自由能减少的方向进行，当自由能达到最小时，系统达到平衡态；

冷却过程——使粒子热运动减弱并渐趋有序，系统能量逐渐下降，从而得到低能的晶体结构。

Metropolis准则

如果用粒子的能量定义材料的状态，Metropolis 算法用一个简单的数学模型描述了退火过程。假设材料在状态i之下的能量为$E(i)$，那么材料在温度T时从状态i进入状态j就遵循如下规律

（1）如果$E(j) \le E(i)$，接受该状态被转换。

（2）如果$E(j) > E(i)$，则状态转换以如下概率被接受

$$e^{\frac{E(i)-E(j)}{KT}},$$

其中K是物理学中的波尔兹曼常数，T是材料温度。

在某一个特定温度下，进行了充分的转换之后，材料将达到热平衡。这时材料处于状态i的概率满足波尔兹曼分布

$$P_T(X=i)=\frac{e^{-\frac{E(i)}{KT}}}{\sum_{j\in S}e^{-\frac{E(j)}{KT}}},$$

其中X表示材料当前状态的随机变量，S表示状态空间集合。

显然

$$\lim_{T\to\infty}\frac{e^{-\frac{E(i)}{KT}}}{\sum_{j\in S}e^{-\frac{E(j)}{KT}}}=\frac{1}{|S|},$$

其中$|S|$表示集合S中状态的数量。这表明所有状态在高温下具有相同的概率。

而当温度下降时，

$$\lim_{T\to 0}\frac{e^{-\frac{E(i)-E_{\min}}{KT}}}{\sum_{j\in S}e^{-\frac{E(j)-E_{\min}}{KT}}}$$

$$=\lim_{T\to 0}\frac{e^{-\frac{E(i)-E_{\min}}{KT}}}{\sum_{j\in S_{\min}}e^{-\frac{E(j)-E_{\min}}{KT}}+\sum_{j\notin S_{\min}}e^{-\frac{E(j)-E_{\min}}{KT}}}$$

$$=\lim_{T\to 0}\frac{e^{-\frac{E(i)-E_{\min}}{KT}}}{\sum e^{-\frac{E(j)-E_{\min}}{KT}}}=\begin{cases}\dfrac{1}{|S_{\min}|}, & \text{若} i\in S_{\min},\\ 0, & \text{其它.}\end{cases}$$

其中 $E_{\min} = \min_{j \in S} E(j)$ 且 $S_{\min} = \{i \mid E(i) = E_{\min}\}$。

上式表明当温度降至很低时，材料会以很大概率进入最小能量状态。

组合优化与物理退火的相似性

相似性比较

组合优化问题	金属物体
解	粒子状态
最优解	能量最低的状态
设定初温	熔解过程
Metropolis抽样过程	等温过程
控制参数的下降	冷却
目标函数	能量

假定要解决的问题是一个寻找最小值的优化问题。将物理学中模拟退火的思想应用于优化问题就可以得到模拟退火寻优方法。

考虑这样一个组合优化问题：优化函数为 $f:x\to R^+$，其中 $x\in S$，它表示优化问题的一个可行解，$R^+=\{y\mid y\in R,y\geq 0\}$，$S$ 表示函数的定义域。$N(x)\subseteq S$ 表示 x 的一个邻域集合。

首先给定一个初始温度T_0和该优化问题的一个初始解$x(0)$，并由$x(0)$生成下一个解$x' \in N(x(0))$，是否接受x'作为一个新解$x(1)$依赖于下面概率

$$P(x(0) \to x') = \begin{cases} 1, & \text{若} f(x') < f(x(0)), \\ e^{-\frac{f(x')-f(x(0))}{T_0}}, & \text{其它.} \end{cases}$$

换句话说，如果生成的解x'的函数值比前一个解的函数值更小，则接受$x(1) = x'$作为一个新解。否则以概率$e^{-\frac{f(x')-f(x(0))}{T_0}}$接受$x'$作为一个新解。

泛泛地说，对于某一个温度T_i和该优化问题的一个解$x(k)$，可以生成x'。接受x'作为下一个新解$x(k+1)$的概率为

$$P(x(k)\to x')=\begin{cases}1, & \text{若} f(x')<f(x(k)),\\ e^{-\frac{f(x')-f(x(k))}{T_i}}, & \text{其它.}\end{cases}$$

在温度T_i下，经过很多次的转移之后，降低温度T_i，得到$T_{i+1} < T_i$。在T_{i+1}下重复上述过程。因此整个优化过程就是不断寻找新解和缓慢降温的交替过程。最终的解是对该问题寻优的结果。

算法描述

1) 随机产生一个初始解x_0，令$x_{best} = x_0$，并计算目标函数值$E(x_0)$;
2) 设置初始温度$T(0)$=To，迭代次数$i = 1$;
3) Do while $T(i) > T_{min}$
   1) for $j = 1 \sim k$
   2) 对当前最优解x_{best}按照某一邻域函数，产生一新的解x_{new}。计算新的目标函数值$E(x_{new})$，并计算目标函数值的增量$\Delta E = E(x_{new}) - E(x_{best})$。
   3) 如果$\Delta E < 0$，则$x_{best} = x_{new}$;
   4) 如果$\Delta E > 0$，则$p = \exp(-\Delta E / T(i))$;
      1) 如果$c = \text{random}[0,1] < p$，$x_{best} = x_{new}$; 否则 $x_{best} = x_{best}$。
   5) End for
4) $i = i + 1$;
5) End Do
6) 输出当前最优点，计算结束。

冷却进度表T(t)

冷却进度表是指从某一高温状态To向低温状态冷却时的降温管理表。

假设时刻t的温度用T(t)来表示，则经典模拟退火算法的降温方式为：

$$T(t)=\frac{T_0}{\lg(1+t)}$$

而快速模拟退火算法的降温方式为：

$$T(t)=\frac{T_0}{1+t}$$

这两种方式都能够使得模拟退火算法收敛于全局最小点。

初始温度T_0

实验表明，初温越大，获得高质量解的几率越大，但花费的计算时间将增加。因此，初温的确定应折衷考虑优化质量和优化效率，常用方法包括:

(1) 均匀抽样一组状态，以各状态目标值的方差为初温。

(2) 随机产生一组状态，确定两两状态间的最大目标值差$|\Delta_{max}|$，然后依据差值，利用一定的函数确定初温。比如，$t_0 = -\Delta_{max}/p_r$，其中p_r为初始接受概率。

(3) 利用经验公式给出。

内循环终止准则

或称Metropolis抽样稳定准则，用于决定在各温度下产生候选解的数目。常用的抽样稳定准则包括:

(1) 检验目标函数的均值是否稳定;

(2) 连续若干步的目标值变化较小;

(3) 按一定的步数抽样。

外循环终止准则

即算法终止准则，常用的包括:

(1) 设置终止温度的阈值;

(2) 设置外循环迭代次数;

(3) 算法搜索到的最优值连续若干步保持不变;

(4) 检验系统熵是否稳定。

模拟退火算法的优缺点

模拟退火算法的优点

质量高；

初值鲁棒性强；

简单、通用、易实现。

模拟退火算法的缺点

由于要求较高的初始温度、较慢的降温速率、较低的终止温度，以及各温度下足够多次的抽样，因此优化过程较长。

应用举例

已知 100 个目标的经度、纬度如表 12.1(表略)所示。我方有一个基地，经度和纬度为（70,40)。假设我方飞机的速度为 1000 公里/小时。我方派一架飞机从基地出发，侦察完所有目标，再返回原来的基地。在每一目标点的侦察时间不计，求该架飞机所花费的时间（假设我方飞机巡航时间可以充分长)。

这是一个旅行商问题。给我方基地编号为 1，目标依次编号为 2，3，…，101，最后我方基地再重复编号为 102（这样便于程序中计算）。距离矩阵 $D=(d_{ij})_{102\times102}$，其中 d_{ij} 表示表示 i,j 两点的距离，$i,j=1,2,\cdots,102$，这里 D 为实对称矩阵。则问题是求一个从点 1 出发，走遍所有中间点，到达点 102 的一个最短路径。

上面问题中给定的是地理坐标（经度和纬度），必须求两点间的实际距离。设A,B两点的地理坐标分别为(x_1,y_1)，(x_2,y_2)，过A,B两点的大圆的劣弧长即为两点的实际距离。以地心为坐标原点O，以赤道平面为XOY平面，以0度经线圈所在的平面为XOZ平面建立三维直角坐标系。

则A,B两点的直角坐标分别为

$$A(R\cos x_1\cos y_1, R\sin x_1\cos y_1, R\sin y_1),$$

$$B(R\cos x_2\cos y_2, R\sin x_2\cos y_2, R\sin y_2),$$

其中$R=6370$为地球半径。A,B两点的实际距离

$$d = R\arccos\left(\frac{\overrightarrow{OA}\cdot\overrightarrow{OB}}{\left|\overrightarrow{OA}\right|\cdot\left|\overrightarrow{OB}\right|}\right),$$

化简得

$$d = R\arccos[\cos(x_1-x_2)\cos y_1\cos y_2+\sin y_1\sin y_2].$$

求解的模拟退火算法描述如下

（1）解空间

解空间 S 可表为 $\{1,2,\cdots,101,102\}$ 的所有固定起点和终点的循环排列集合，即

$$S=\{(\pi_1,\cdots,\pi_{102})\,|\,\pi_1=1,(\pi_2,\cdots,\pi_{101})\text{为}\{2,3,\cdots,101\}\text{的循环排列},\pi_{102}=102\}$$

其中每一个循环排列表示侦察 100 个目标的一个回路，$\pi_i=j$ 表示在第 $i-1$ 次侦察目标 j，初始解可选为 $(1,2,\cdots,102)$，本文中我们先使用 Monte Carlo（蒙特卡洛）方法求得一个较好的初始解。

（2）目标函数

目标函数（或称代价函数）为侦察所有目标的路径长度。要求

$$\min f(\pi_1,\pi_2,\cdots,\pi_{102})=\sum_{i=1}^{101} d_{\pi_i\pi_{i+1}},$$

而一次迭代由下列三步构成

（3）新解的产生

设上一步迭代的解为

$$\pi_1 \cdots \pi_{u-1}\pi_u\pi_{u+1} \cdots \pi_{v-1}\pi_v\pi_{v+1} \cdots \pi_{w-1}\pi_w\pi_{w+1} \cdots \pi_{102}。$$

i） 2 变换法

任选序号u,v，交换u与v之间的顺序，变成逆序，此时的新路径为

$$\pi_1 \cdots \pi_{u-1}\pi_v\pi_{v-1} \cdots \pi_{u+1}\pi_u\pi_{v+1} \cdots \pi_{102}.$$

ii） 3 变换法

任选序号u,v和w，将u和v之间的路径插到w之后，对应的新路径为

$$\pi_1\cdots\pi_{u-1}\pi_{v+1}\cdots\pi_w\pi_u\cdots\pi_v\pi_{w+1}\cdots\pi_{102}.$$

（4）代价函数差

对于 2 变换法，路径差可表示为

$$\Delta f = (d_{\pi_{u-1}\pi_v} + d_{\pi_u\pi_{v+1}}) - (d_{\pi_{u-1}\pi_u} + d_{\pi_v\pi_{v+1}}).$$

（5）接受准则

$$P=\begin{cases}1, & \Delta f<0,\\ \exp(-\Delta f/T), & \Delta f\geq 0.\end{cases}$$

如果$\Delta f<0$，则接受新的路径。否则，以概率$\exp(-\Delta f/T)$接受新的路径，即用计算机产生一个[0,1区间上均匀分布的随机数 rand，若 rand$\leq\exp(-\Delta f/T)$则接受。

（6）降温

利用选定的降温系数α进行降温，取新的温度T为αT（这里T为上一步迭代的温度），这里选定$\alpha = 0.999$。

(7) 结束条件

用选定的终止温度$e=10^{-30}$，判断退火过程是否结束。若$T<e$，算法结束，输出当前状态。

计算结果为 44 小时左右。其中的一个巡航路径如图 12.1 所示。

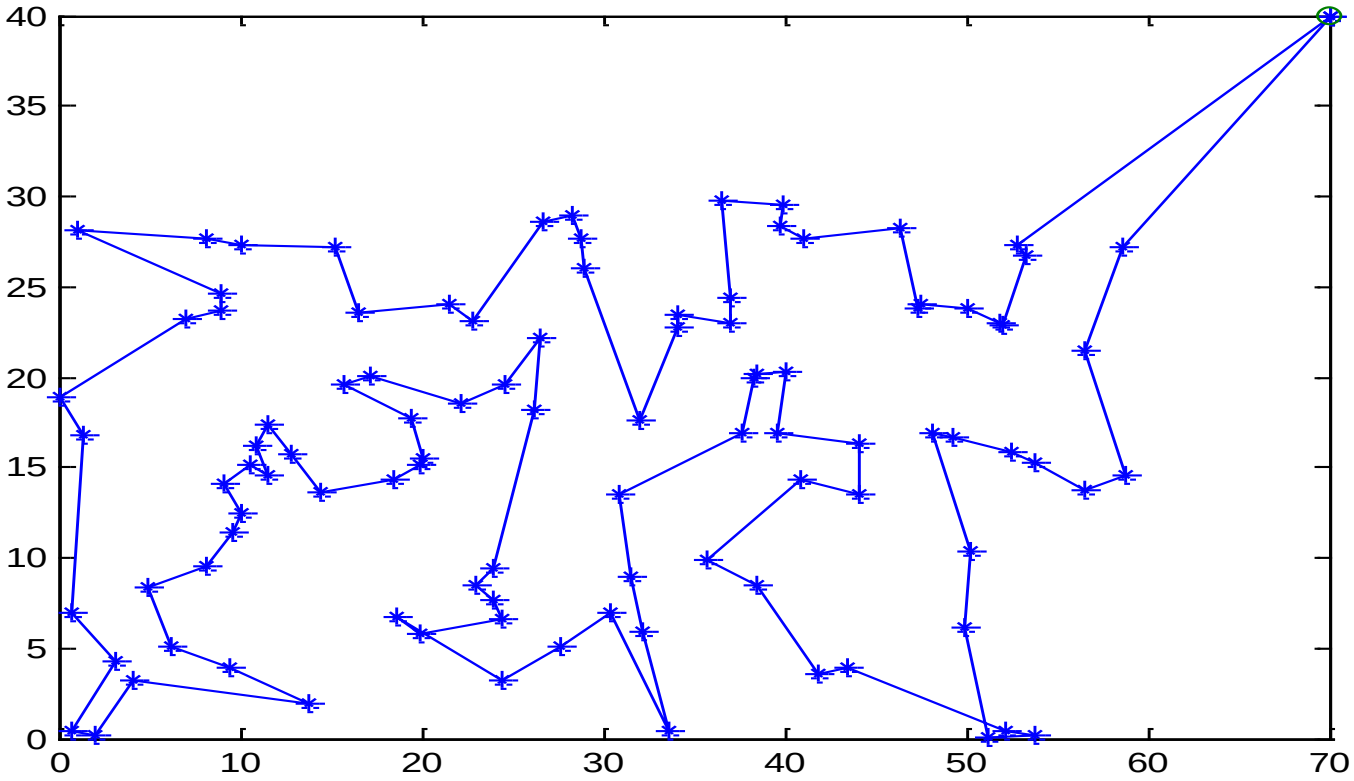


图 12.1　模拟退火算法求得的巡航路径示意图

算法改进

- （1）设计合适的状态产生函数，使其根据搜索进程的需要
- 表现出状态的全空间分散性或局部区域性。
- （2）设计高效的退火策略。
- （3）避免状态的迂回搜索。
- （4）采用并行搜索结构。
- （5）为避免陷入局部极小，改进对温度的控制方式
- （6）选择合适的初始状态。
- （7）设计合适的算法终止准则。

算法改进

也可通过增加某些环节而实现对模拟退火算法的改进。包括：

(1) 增加升温或重升温过程。在算法进程的适当时机，将温度适当提高，从而可激活各状态的接受概率，以调整搜索进程中的当前状态，避免算法在局部极小解处停滞不前。

(2) 增加记忆功能。为避免搜索过程中由于执行概率接受环节而遗失当前遇到的最优解，可通过增加存储环节，将“Best So Far”的状态记忆下来。

(3) 增加补充搜索过程。即在退火过程结束后，以搜索到的最优解为初始状态，再次执行模拟退火过程或局部性搜索。

(4) 对每一当前状态，采用多次搜索策略，以概率接受区域内的最优状态，而非标准SA的单次比较方式。

(5) 结合其他搜索机制的算法，如遗传算法、混沌搜索等。

(6) 上述各方法的综合应用。

Part 3　遗传算法

遗传算法（Genetic Algorithm）

- 进化算法(Evolutionary Algorithm)

遗传算法（Genetic Algorithm）

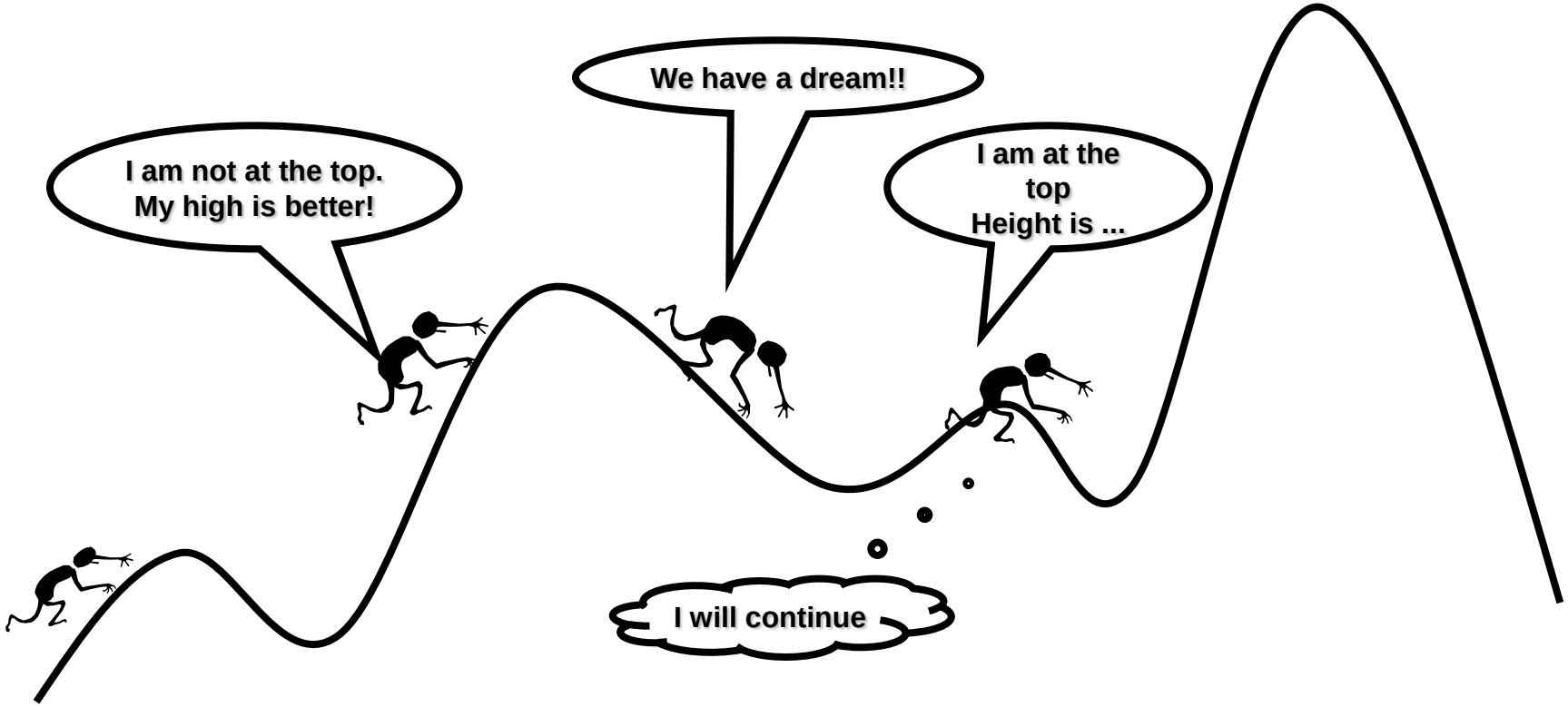
- Darwin(1859): “物竞天择，适者生存”
- John Holland (university of Michigan, 1975)
  《Adaptation in Natural and Artificial System》
- 遗传算法作为一种有效的工具，已广泛地应用于最优化问题求解之中。
- 遗传算法是一种基于自然群体遗传进化机制的自适应全局优化概率搜索算法。它摒弃了传统的搜索方式，模拟自然界生物进化过程，采用人工的方式对目标空间进行随机化搜索。

遗传算法的搜索机制

遗传算法模拟自然选择和自然遗传过程中发的繁殖、交叉和基因突变现象，在每次迭代中都保留一组候选解，并按某种指标从解群中选取较优的个体，利用遗传算子(选择、交叉和变异)对这些个体进行组合，产生新一代的候选解群，重复此过程，直到满足某种收敛指标为止。

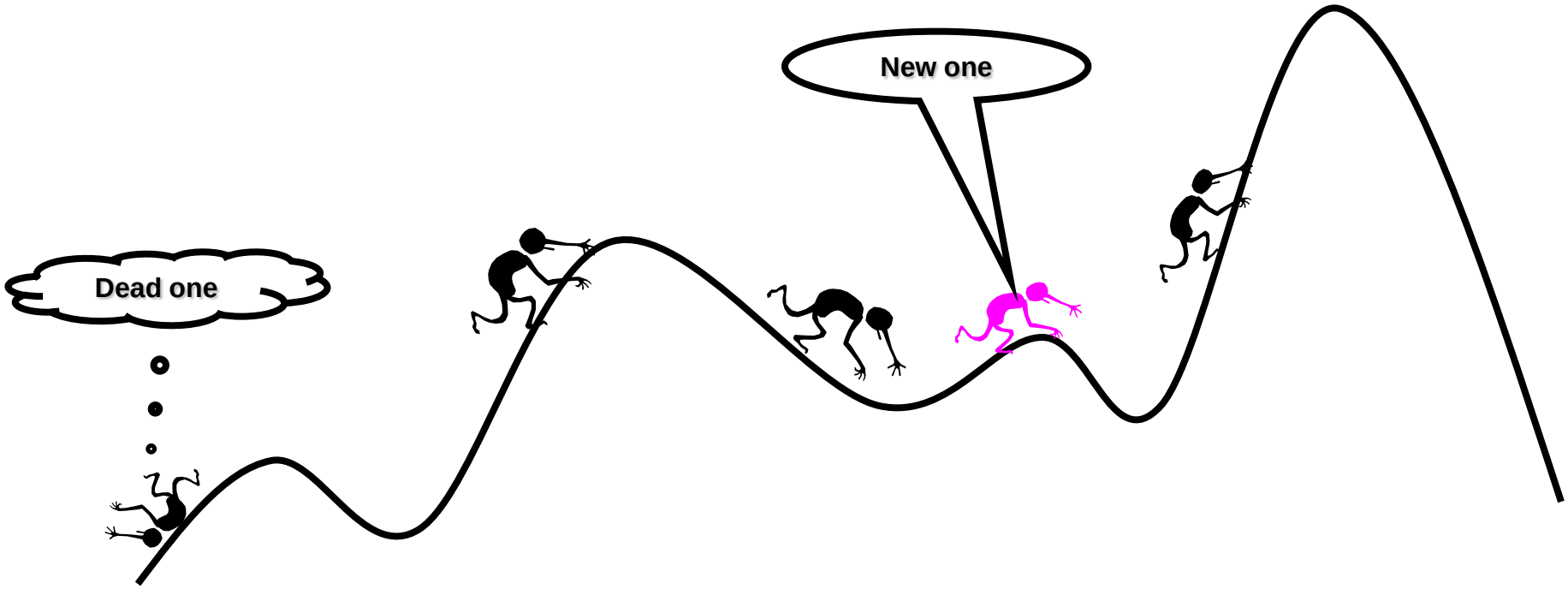
遗传算法（GA）

GA-----第0代



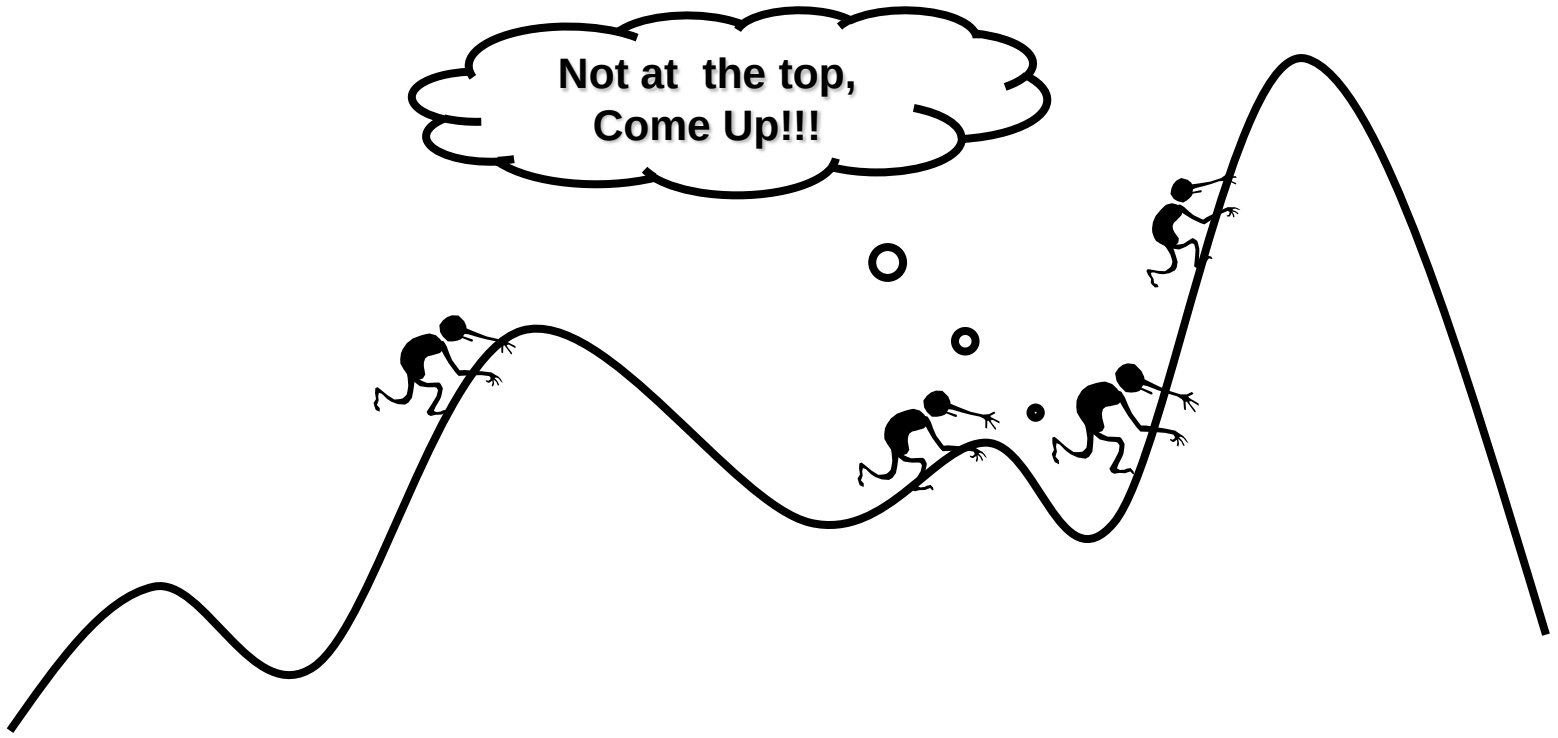
遗传算法（GA）

GA----第1代



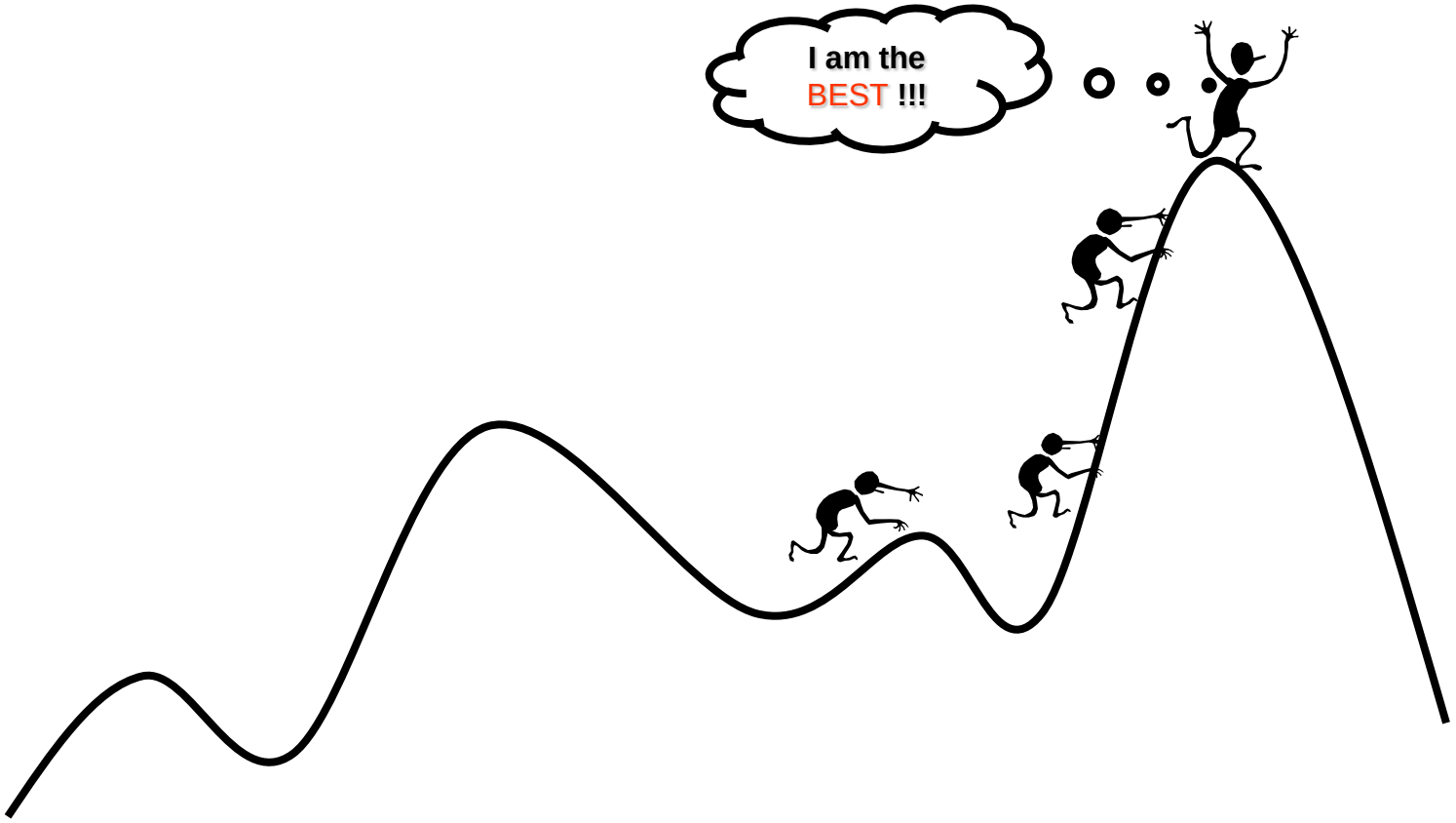
遗传算法（GA）

GA----第?代



遗传算法（GA）

GA----第N代



基本遗传算法（SGA）

基本遗传算法（Simple Genetic Algorithms, 简称SGA）是一种统一的最基本的遗传算法，它只使用选择、交叉、变异这三种基本遗传算子，其遗传进化操作过程简单，容易理解，是其他一些遗传算法的雏形和基础，它不仅给各种遗传算法提供了一个基本框架，同时也具有一定的应用价值。

实例 1： 函数最值

求函数$f(x)=x^2$的最大值，x为自然数且$0 \le x \le 31$.

SGA参数:

- ✓ 编码方式：二进制码

  e. g. 00000↔0； 01101 ↔ 13； 11111↔31

- ✓ 种群规模：4
- ✓ 随机初始群体
- ✓ “转盘赌”选择
- ✓ 一点杂交，二进制变异

手工方式完成演示SGA过程

初始

编号	初始群体	变量x	适应值$f(x)=x^2$	选择比	期望	实际数目
1	0 1 1 0 1	13	169	0.14	0.58	1
2	1 1 0 0 0	24	576	0.49	1.97	2
3	0 1 0 0 0	8	64	0.06	0.22	0
4	1 0 0 1 1	19	361	0.31	1.23	1
和			1170	1.00	4.00	4
平均值			293	0.25	1.00	1
最大值			576	0.49	1.97	2

交叉

编号	交配池	交叉位置	交叉后 新的种群	变量x	适应值 $f(x)=x^2$
1	0 1 1 0 | 1	4	0 1 1 0 0	12	144
2	1 1 0 0 | 0	4	1 1 0 0 1	25	625
2	1 1 | 0 0 0	2	1 1 0 1 1	27	729
4	1 0 | 0 1 1	2	1 0 0 0 0	16	256
和					1754
平均值					439
最大值					729

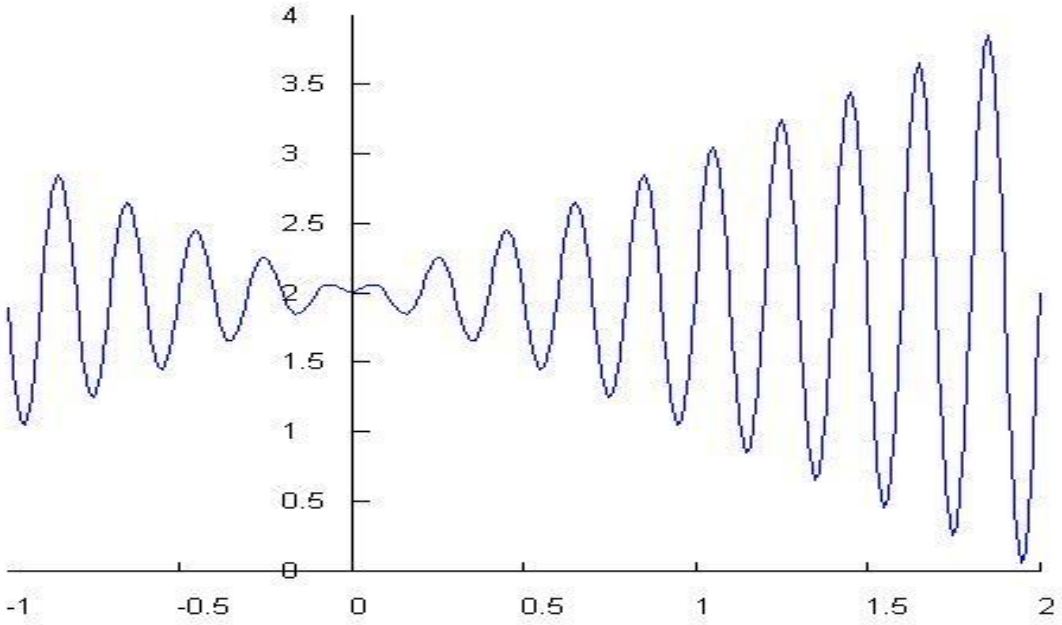
变异

编号	交叉后 新的种群	变异后 新的种群	变量x	适应值 $f(x)=x^2$
1	0 1 1 0 0	1 1 1 0 0	26	676
2	1 1 0 0 1	1 1 0 0 1	25	625
2	1 1 0 1 1	1 1 0 1 1	27	729
4	1 0 0 0 0	1 0 1 0 0	18	324
和 平均值 最大值				2354 588.5 729

实例 2：连续函数的最值

求下列函数的最大值：

$$f(x) = x\sin(10\pi x) + 2.0 \qquad x \in [-1, 2]$$



编码

➢编码

✓ 实数问题：变量z为实数，如何把

$$z \in [x,y] \longrightarrow \{a_1,\ldots,a_L\} \in \{0,1\}^L$$

✓ $[x,y] \rightarrow \{0,1\}^L$ 必须可逆(一个表现型对应一个基因型)

✓ 解码算子：$\Gamma: \{0,1\}^L \rightarrow [x,y]$

$$\Gamma(b_L,\ldots,a_0) = x + \frac{y-x}{2^L-1} \cdot \left(\sum_{i=0}^{L-1} b_i \cdot 2^i\right) \in [x,y]$$

✓ 染色体长度L决定可行解的最大精度

✓ 高精度 ⟺ 长染色体(慢进化)

编码

设定求解精确到6位小数，因区间长度位2-(-1)=3,则需将区间分为$3X10^6$等份。因 $2097152=2^{21}< 3X10^6\leq 2^{22}=4194304$。故编码的二进制串长L=22。

将一个二进制串$(b_{21}b_{20}\ldots b_0)$转化为10进制数：

$$(b_{21},\ldots,b_0)_2=-1+\frac{2-(-1)}{2^L-1}\cdot(\sum_{i=0}^{21}b_i\cdot 2^i)_{10}\in[-1,2]$$

e.g. <0000000000000000000000> ↔ -1;

<1111111111111111111111> ↔ 2

<1110000000111111000101> ↔ 1.627 888

1.627888 = -1+3x$(1110000000111111000101)_2/(2^{22}-1)$

= -1+3x3674053/(222-1)

初始化种群，适应函数

- 随机初始化种群
- 适应函数

本实例目标函数在定义域内均大于0，且是求函数最大值，故直接引用目标函数作为适应函数：

$$f(s) = f(x)$$

其中二进制串s对于变量x的值。

e.g. s_1 =<000000111000000010000> ↔ x_1= -0.958 973

适应值: $f(s_1) = f(x_1)$ =1.078 878

s_2=<111000000111111000101> ↔ x_2= 1.627 888

适应值: $f(s_2) = f(x_2)$ = 3.250 650

遗传

- 选择操作（“轮盘赌”选择）
- 交叉操作（单点交叉）

交叉前(父): s_1=<00000 | 0111000000010000>

s_2=<11100 | 0000011111000101>

交叉后(子): s'_1=<00000 | 0000011111000101>

s'_2=<11100 | 0111000000010000>

适应值: $f(s'_1) = f(-0.998\ 113) = 1.940\ 865$

$f(s'_2) = f(1.666\ 028) = 3.459\ 245$

s'_2的适应值比其双亲个体的适应值高。

变异

- **变异操作**

  变异前(父): s_2=<111000000011111000101>

  变异后(子): s'_2=<111010000011111000101>

  适应值 $f(s'_2) = f(1.721\ 638) = 0.917\ 743$ 比 $f(s_2)$小

  变异前(父): s_2=<111000000011111000101>

  变异后(子): s''_2=<111000000111111000101>

  适应值 $f(s''_2) = f(1.630\ 818) = 3.343\ 555$ 比 $f(s_2)$大

变异操作有”扰动”作用，同时具有增加种群多样性的效果

模拟结果

遗传算法的参数：

种群规模： 50

染色体长度： L=22

最大进化代数： 150

交叉概率： P_c=0.25

变异概率： P_m=0.01

模拟结果（最佳个体进化情况）

世代数	染色体编码	变量x	适应值
1	100011100001011000111 1	1.831 624	3.534 806
4	000001101100010100111 1	1.842 416	3.790 362
11	011010101110011100111 1	1.854 860	3.833 286
17	111010101111111010011 1	1.847 536	3.842 004
34	110000110111101100111 1	1.853 290	3.843 402
40	110100100010001100111 1	1.848 443	3.846 232
54	100011011010001100111 1	1.848 699	3.847 155
71	010011011000101100111 1	1.850 897	3.850 162
89	110100111111001100111 1	1.850 549	3.850 274
150	110100111111001100111 1	1.850 549	3.850 274

遗传算法的思路与特点

自组织、自适应和自学习性

在编码方案、适应度函数及遗传算子确定后，算法将利用进化过程中获得的信息自行组织搜索。

本质并行性

内在并行性与内含并行性

不需求导

只需目标函数和适应度函数

概率转换规则

强调概率转换规则，而不是确定的转换规则

遗传算法的基本操作

选择

适应度计算：

- 按比例的适应度函数（proportional fitness assignment）
- 基于排序的适应度计算（Rank-based fitness assignment

选择算法：

- 轮盘赌选择（roulette wheel selection）
- 随机遍历抽样（stochastic universal selection）
- 局部选择（local selection）
- 截断选择（truncation selection）
- 锦标赛选择（tournament selection）

遗传算法的基本操作

交叉或基因重组

实值重组（real valued recombination）：

- 离散重组（discrete recombination）
- 中间重组（intermediate recombination）
- 线性重组（linear recombination）
- 扩展线性重组（extended linear recombination）

遗传算法的基本操作

交叉或基因重组

二进制交叉（binary valued crossover）：

- 单点交叉（single-point crossover）
- 多点交叉（multiple-point crossover）
- 均匀交叉（uniform crossover）
- 洗牌交叉（shuffle crossover）
- 缩小代理交叉（crossover with reduced surrogate）

遗传算法的基本操作

变异

- 实值变异
- 二进制变异

为便于计算，一般来说，每一代群体的个体数目都取相等。群体规模越大、越容易找到最优解，但由于受到计算机的运算能力的限制，群体规模越大，计算所需要的时间也相应地增加。进化终止条件指的是当进化到什么时候结束，它可以设定到某一代进化结束，也可以根据找出近似最优解是否满足精度要求来确定。下表列出了生物遗传概念在遗传算法中的对应关系。

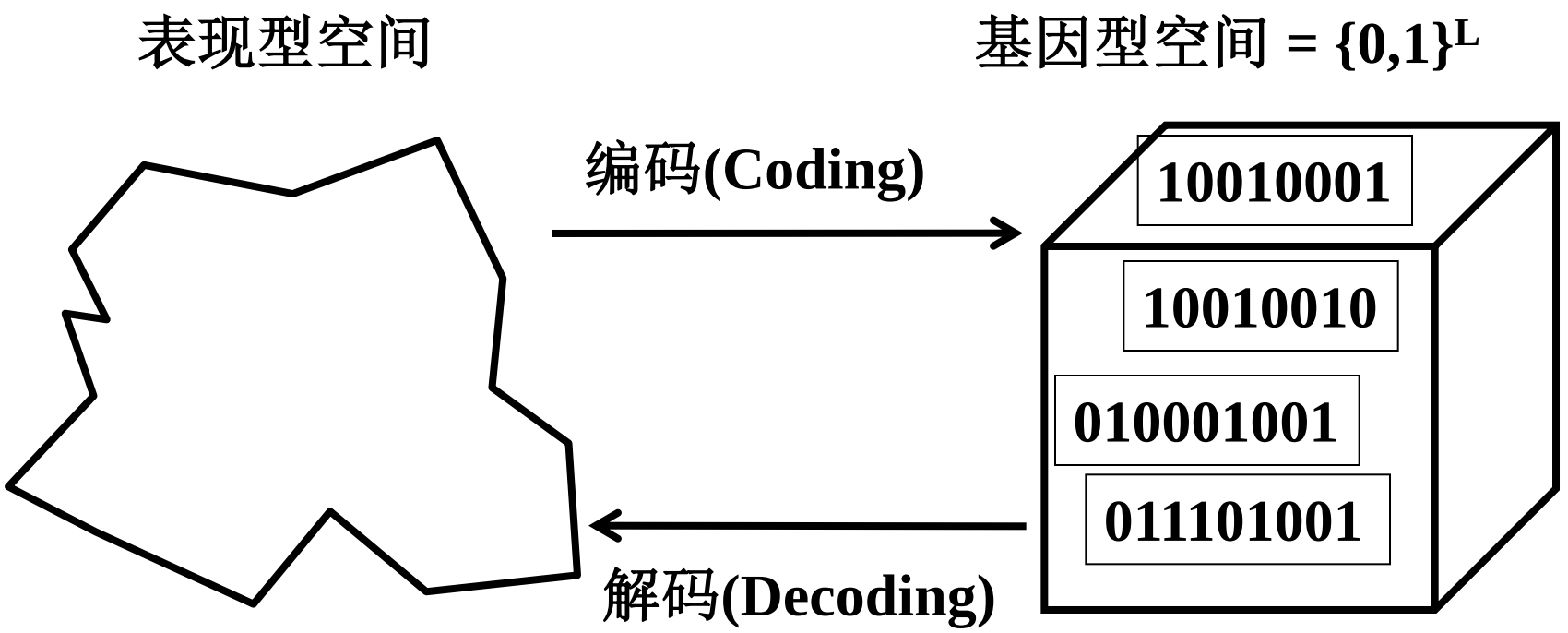
生物进化与遗传算法对应关系

生物进化		遗传算法
环境	→	适应函数
适者生存	→	适应函数值最大的解被保留的概率最大
个体	→	问题的一个解
染色体	→	解的编码
基因	→	编码的元素
群体	→	被选定的一组解
种群	→	根据适应函数选择的一组解
交叉	→	以一定的方式由双亲产生后代的过程
变异	→	编码的某些分量发生变化的过程

如何设计遗传算法

- 如何进行编码？
- 如何产生初始种群？
- 如何定义适应函数？
- 如何进行遗传操作(复制、交叉、变异)？
- 如何产生下一代种群？
- 如何定义停止准则？

编码


表现型空间
基因型空间 = {0,1}^L
编码(Coding)
10010001
10010010
010001001
011101001
解码(Decoding)

选择

- 选择(复制)操作把当前种群的染色体按与适应值成正比例的概率复制到新的种群中
- 主要思想: 适应值较高的染色体体有较大的选择(复制)机会
- 实现1：”轮盘赌”选择(Roulette wheel selection)
  - ✓将种群中所有染色体的适应值相加求总和，染色体适应值按其比例转化为选择概率Ps
  - ✓产生一个在0与总和之间的的随机数m
  - ✓从种群中编号为1的染色体开始，将其适应值与后续染色体的适应值相加，直到累加和等于或大于m

选择

设种群的规模为N

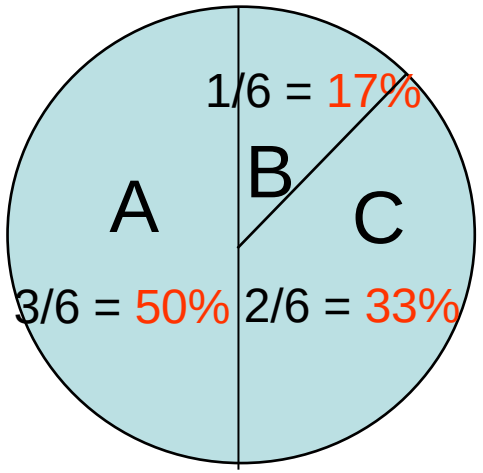
x_i是i为种群中第i个染色体

染色体x_i被选概率

$$p_s(x_i) = \frac{F(x_i)}{\sum_{j=1}^{N} F(x_j)}$$

$f(x_i)$ 第i个染色体的适应度值

$\sum f(x_i)$ 种群中所有染色体适应度值之和。



fitness(A) = 3

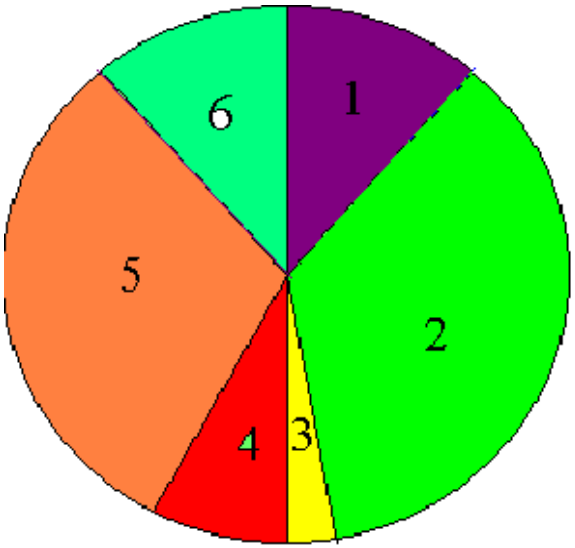
fitness(B) = 1

fitness(C) = 2

选择

染色体的适应值和所占的比例

序号	染色体	适应度值	所占比例	累计
1	01110	8	16	8
2	11000	15	30	23
3	00100	2	4	25
4	10010	5	10	30
5	01100	12	24	42
6	00011	8	16	50



轮盘赌选择

选择

染色体被选的概率

染色体编号	1	2	3	4	5	6
染色体	01110	11000	00100	10010	01100	00011
适应度	8	15	2	5	12	8
被选概率	0.16	0.3	0.04	0.1	0. 24	0.16
适应度累计	8	23	25	30	42	50

被选的染色体

随机数	23	49	13	38	6	27
所选号码	2	6	2	5	1	4
所选染色体	11000	00011	11000	01100	01110	10010

选择

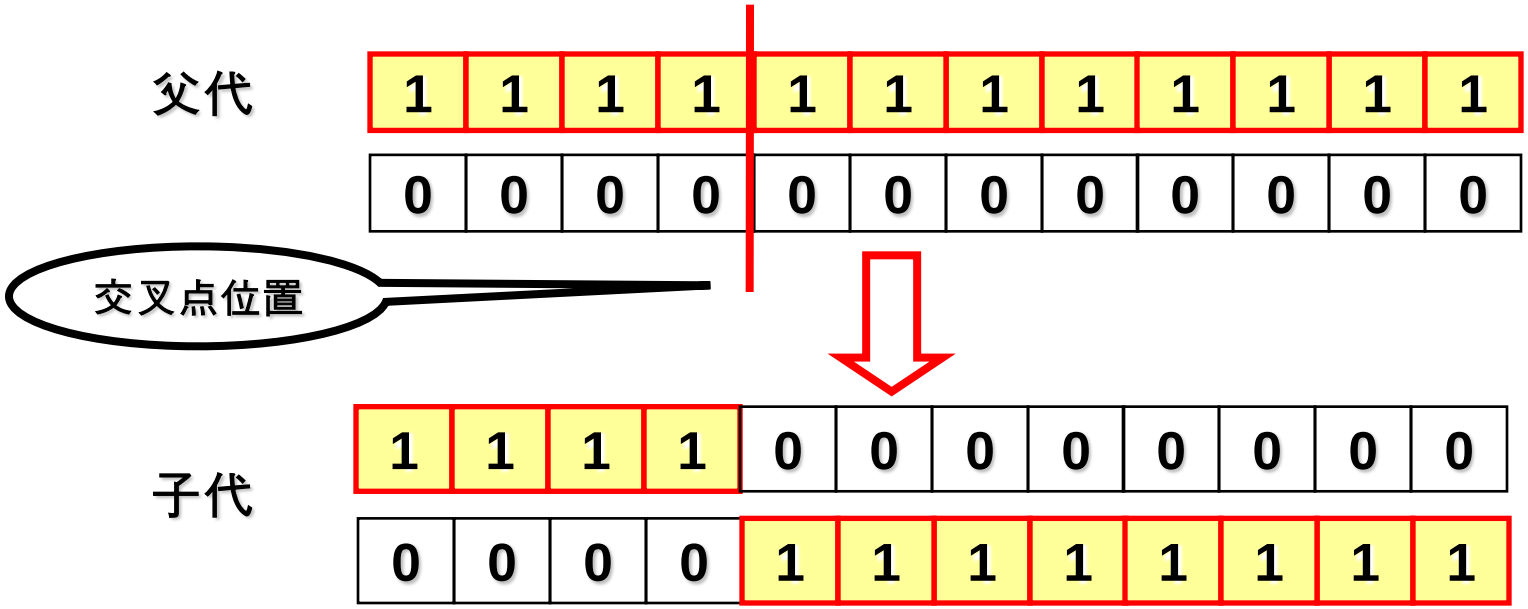
- 其他选择法：
  - ✓ 随机遍历抽样(Stochastic universal sampling)
  - ✓ 局部选择(Local selection)
  - ✓ 截断选择(Truncation selection)
  - ✓ 竞标赛选择(Tournament selection)
- 特点：选择操作得到的新的群体称为交配池，交配池是当前代和下一代之间的中间群体，其规模为初始群体规模。选择操作的作用效果是提高了群体的平均适应值(低适应值个体趋于淘汰，高适应值个体趋于选择)，但这也损失了群体的多样性。选择操作没有产生新的个体，群体中最好个体的适应值不会改变。

交叉

- 遗传交叉(杂交、交配、有性重组)操作发生在两个染色体之间，由两个被称之为双亲的父代染色体，经杂交以后，产生两个具有双亲的部分基因的新的染色体，从而检测搜索空间中新的点。
- 选择(复制)操作每次作用在一个染色体上，而交叉操作每次作用在从交配池中随机选取的两个个体上(交叉概率P_c)。
- 交叉产生两个子染色体，他们与其父代不同，且彼此不同， 每个子染色体都带有双亲染色体的遗传基因。

单点交叉

- 在双亲的父代染色体中随机产生一个交叉点位置
- 在交叉点位置分离双亲染色体
- 互换交叉点位置右边的基因码产生两个子代染色体
- 交叉概率P_c一般范围为(60%, 90%)，平均约80%



交叉

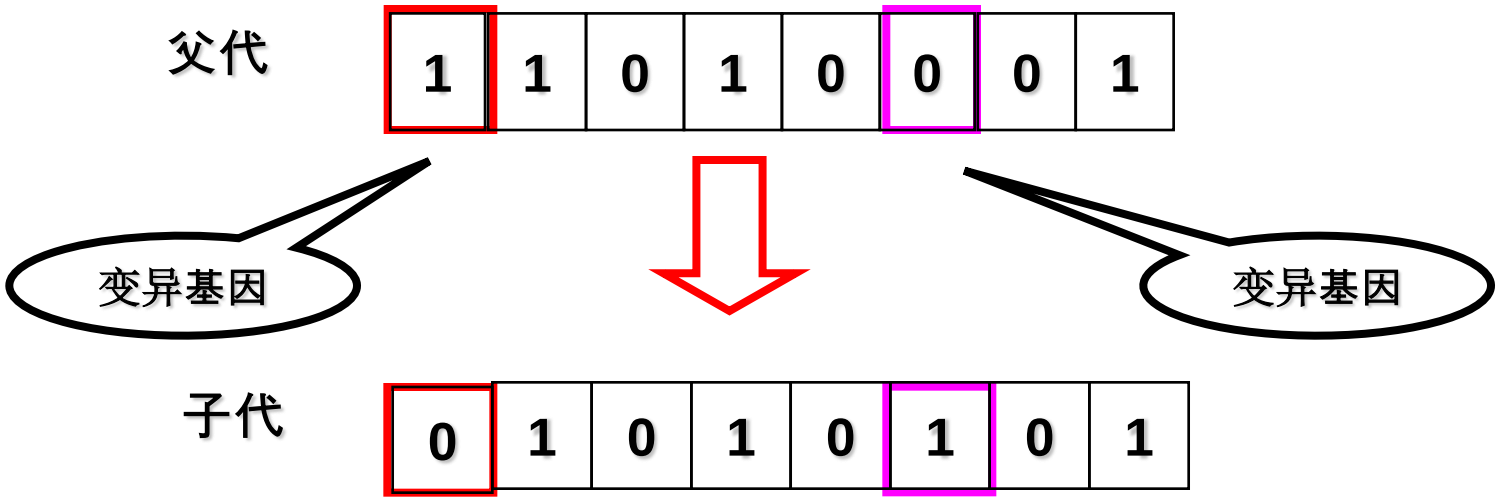
- 单点交叉操作可以产生与父代染色体完全不同的子代染色体；它不会改变父代染色体中相同的基因。但当双亲染色体相同时，交叉操作是不起作用的。

假如交叉概率P_c＝50%,则交配池中50%的染色体(一半染色体)将进行交叉操作，余下的50%的染色体进行选择(复制)操作。

- GA利用选择和交叉操作可以产生具有更高平均适应值和更好染色体的群体

变异

- 以变异概率P_m改变染色体的某一个基因,当以二进制编码时，变异的基因由0变成1，或者由1变成0。
- 变异概率P_m 一般介于1/种群规模与1/染色体长度之间，平均约1-2%



变异

- 比起选择和交叉操作，变异操作是GA中的次要操作，但它在恢复群体中失去的多样性方面具有潜在的作用。

在GA执行的开始阶段，染色体中一个特定位上的值1可能与好的性能紧密联系，即搜索空间中某些初始染色体在那个位上的值1可能一致产生高的适应值。因为越高的适应值与染色体中那个位上的值1相联系，选择操作就越会使群体的遗传多样性损失。等到达一定程度时，值0会从整个群体中那个位上消失，然而全局最优解可能在染色体中那个位上为0。如果搜索范围缩小到实际包含全局最优解的那部分搜索空间，在那个位上的值0就可能正好是到达全局最优解所需要的。

适应函数

- GA在搜索中不依靠外部信息，仅以适应函数为依据，利用群体中每个染色体(个体)的适应值来进行搜索。以染色体适应值的大小来确定该染色体被遗传到下一代群体中的概率。染色体适应值越大，该染色体被遗传到下一代的概率也越大；反之，染色体的适应值越小，该染色体被遗传到下一代的概率也越小。因此适应函数的选取至关重要，直接影响到GA的收敛速度以及能否找到最优解。
- 群体中的每个染色体都需要计算适应值
- 适应函数一般由目标函数变换而成

适应函数

➢适应函数常见形式：

✓直接将目标函数转化为适应函数

- 若目标函数为最大化问题：

$$Fitness(f(x)) = f(x)$$

- 若目标函数为最小化问题：

$$Fitness(f(x)) = -f(x)$$

•缺点：

(1) 可能不满足轮盘赌选择中概率非负的要求

(2) 某些代求解的函数值分布上相差很大，由此得到的评价适应值可能不利于体现群体的评价性能，影响算法的性能。

停止准则

- 种群中个体的最大适应值超过预设定值
- 种群中个体的平均适应值超过预设定值
- 种群中个体的进化代数超过预设定值

基本步骤

(1) 随机产生初始种群;

(2) 计算种群体中每个个体的适应度值,判断是否满足停止条件，若不满足，则转第(3)步,否则转第(6)步;

(3) 按由个体适应值所决定的某个规则选择将进入下一代的个体;

(4) 按交叉概率P_c进行交叉操作,生产新的个体;

(5) 按变异概率P_m进行变异操作,生产新的个体;

(6) 输出种群中适应度值最优的染色体作为问题的满意解或最优解。

组合最优化问题

典型问题：

巡回旅行商问题(Traveling Salesman Problem)

作业调度问题(Job Shop Scheduling Problem)

背包问题(Knapsack Problem)

图着色问题… … …

巡回旅行商问题（TSP）

TSP，也称货郎担问题，是一个NP完全问题。

TSP描述：

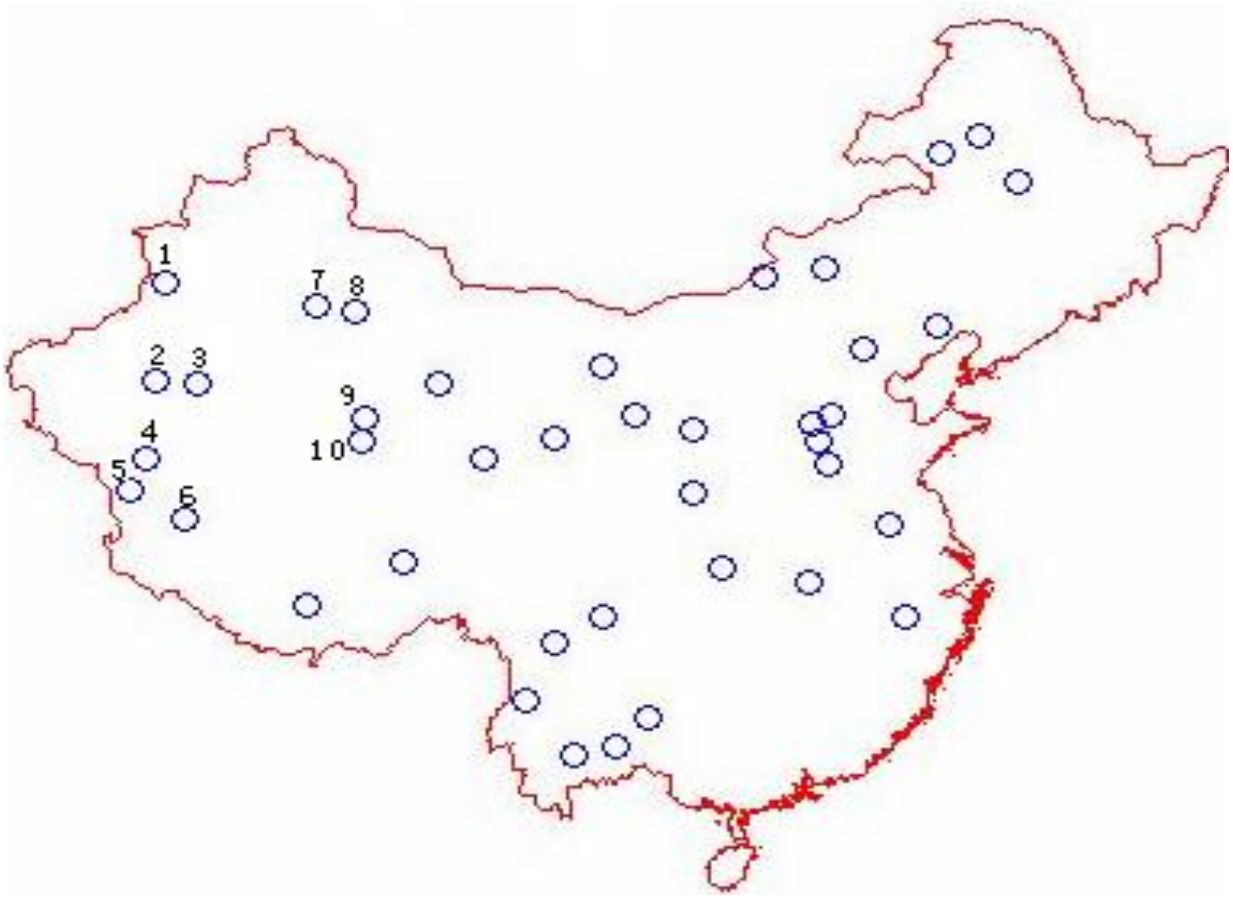
- 图论:设图G=(V,E),其中V是顶点集，E是边集。设$C=(c_{ij})$是与E相联系的距离矩阵。寻找一条通过所有顶点且每个顶点只通过一次的最短距离回路(Hamilton回路)。实际应用中，C也可解释为费用或旅行时间矩阵。

- 实际:一位推销员从自己所在城市出发，必须遍访所有城市之后又回到原来的城市，求使其旅行费用最少的路径。

巡回旅行商问题（TSP）

中国货郎担问题:

- 城市数: 40
- 城市编号1,2,…,40
- 寻找一条最短路径



TSP复杂性

搜索空间庞大

TSP涉及求多个变量的函数的最小值，求解很困难。其可能的路径条数随着城市数目n成指数增长，如，5个城市对应12条路径；10个城市对应181 440条路径；100个城市对应4.6663×10^{155}条路径。如此庞大的搜索空间，常规解法和计算工具都遇到计算上的困难。只能寻找近似解法，如神经网络方法、模拟退火法、遗传算法等。

TSP编码：路径表示

染色体表示成所有城市的一个排列，若有n个城市，一条可能路径编码为长度为n的整数向量($i_1,i_2,\ldots,i_n$),其中i_k表示第i_k个城市。

例如: 路径编码向量(5 1 7 8 9 4 6 2 3)直接表示一条旅行路程(5->1->7->8->9->4->6->2->3)。

此向量是1到n的一个排列，即从1到n的每个整数在这个向量中正好出现一次,不能有重复。这样，基本遗传算法的基因操作生成的个体不能满足这一约束条件，需寻求其他遗传操作。

TSP交叉

一般的交叉操作会产生不合适的解，如

需其他方式的交叉(重组)操作,

如部分匹配交叉(Partially Matched Crossover,PMX)、

顺序交叉(Ordered Crossover,OX)、

循环交叉(Cycle Crossover,CX)、

边重组(Edge Recombination)。

TSP交叉 1： 部分匹配交叉

- 双亲P1,P2随机选取两个交叉点，得到一个匹配段,根据交叉点中间段给出映射关系。

P1	1	2	3	4	5	6	7	8	9
P2	9	3	7	8	2	6	5	1	4

映射关系：

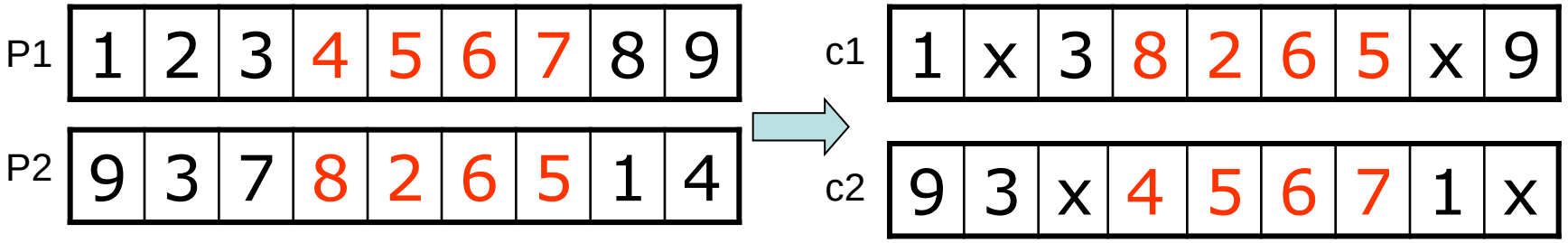
4 ↔8、 5 ↔2、 7 ↔5

- 交换两个交叉点之间的编码,(X表示未定码)

c1	x	x	x	8	2	6	5	x	x
c2	x	x	x	4	5	6	7	x	x

TSP交叉 1： 部分匹配交叉

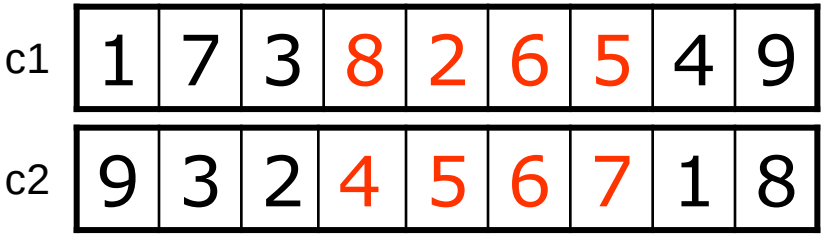
- 子个体C_1，C_2分别从其父个体中继承未映射城市码



- 再根据映射关系，对以上未定码，使用最初双亲码，得到子个体的对应码。映射关系存在传递关系，则选择未定码交换。

映射关系：

4 ↔8、5 ↔2、7 ↔5



TSP交叉 2 ： 顺序交叉

- 双亲**P1,P2**随机选取两个交叉点

P1	1	2	3	4	5	6	7	8	9
P2	9	3	7	8	2	6	5	1	4

- 两个交叉点间的中间段保存不变

c1	x	x	x	4	5	6	7	x	x
c2	x	x	x	8	2	6	5	x	x

- 子个体C1的未定码的确定需要父个体P2的未选定城市码，子个体C2的未定码的确定需要父个体P1的未选定城市码。

TSP交叉 2 ： 顺序交叉

- 记取父个体P_2从第二个交叉点开始城市码的排列顺序，当到达表尾时，返回表头继续记录，直到第二个交叉点。

P2

9	3	7	8	2	6	5	1	4

c1

x	x	x	4	5	6	7	x	x

- 得到父个体P2的排列顺序1-4-9-3-7-8-2-6-5,并将C1已有城市码4,5,6,7从中去掉，得到排列顺序1-9-3-8-2，再将此顺序复制到C1，复制点也是从第二个交叉点开始，得到C1。

c1

3	8	2	4	5	6	7	1	9

同理的C2,

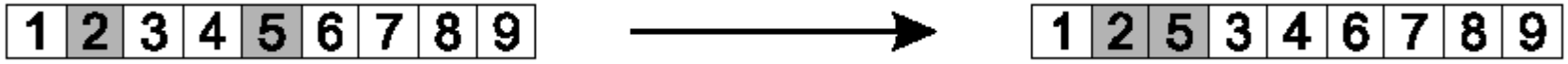
c2

3	4	7	8	2	6	5	9	1

TSP变异

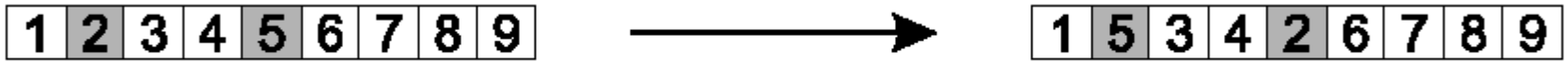
➢ Insert Mutation

随机选取个体中两个编码，然后把第二个编码放在第一个编码之后，其他编码顺次调节位置。



➢ Swap mutation

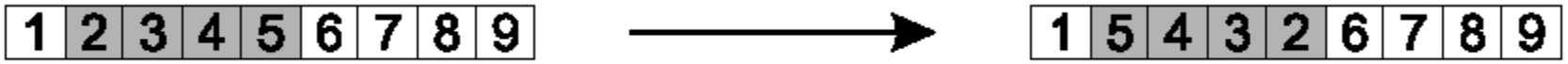
随机选取个体中两个编码，然后交换它们的位置。



TSP变异

- **Inversion mutation**

随机选取个体中一段编码，然后颠倒这段编码的顺序。



- **Scramble mutation**

随机选取个体上一段编码,然后打乱这段编码的顺序。

选取的编码不一定是邻接编码

TSP的GA过程

- 从N个随机起点开始产生N条路径，N为种群的规模;
- 利用最优方法搜索每条路径的局部最优解;
- 选择交叉对使在平均性能之上的个体得到更多的子代;
- 交叉和变异;
- 搜索每条路径得到其极小解，如果不收敛，则回到第3步；否则，停止。

GA的MATLAB实现

软件平台(Software Platforms):

- MATLAB 7.x
  - ✓ Genetic Algorithm and Direct Search Toolbox 2.0.1
- MATLAB 6.x(or 7.x)+GAOT
  - ✓ GAOT: Genetic Algorithm Optimization Toolbox
  - ✓ 美国North Carolina State University开发
- MATLAB 6.x(or 7.x)+GEATbx
  - ✓ GEATbx: Genetic and Evolutionary Algorithm Toolbox
  - ✓ 英国The University of Sheffield开发
  - ✓ 《MATLAB遗传算法工具箱及应用》(雷英杰等,西安电子科技大学出版社,2005)基于此工具箱

GA工具箱

核心函数：

(1) [pop]=**initializega**(num,bounds,eevalFN,eevalOps,options)

------初始种群的生成函数

【输出参数】

pop-----生成的初始种群

【输入参数】

num-----种群中的个体数目

bounds-----代表变量的上下界的矩阵

eevalFN-----适应度函数

eevalOps-----传递给适应度函数的参数

options-----选择编码形式(浮点编码或是二进制编码)与精度，如 [type prec]，

type-----为1时选择浮点编码，否则为二进制编码

prec-----变量进行二进制编码时指定的精度，默认[1e-6 1]

GA工具箱

(2) [x,endPop,bPop,traceInfo] =
ga(bounds,evalFN,evalOps,startPop,opts,termFN,termOps,selectFN,…
selectOps,xOverFNs,xOverOps,mutFNs,mutOps)
-------遗传算法函数

【输出参数】

x------求得的最优解

endPop------最终得到的种群

bPop------最优种群的一个搜索轨迹

traceInfo------每代种群中最优及平均个体构成的矩阵

【输入参数】

bounds------代表变量上下界的矩阵

evalFN------适应度函数

evalOps------传递给适应度函数的参数

startPop------初始种群

GA工具箱

【输入参数】

opts------- [epsilon prob_ops display]，opts(1:2)等同于initializega的options参数，第三个参数控制是否输出，一般为0。如[1e-6 1 0]

termFN-------终止函数的名称,如['maxGenTerm']

termOps-------传递个终止函数的参数,如[100]

selectFN-------选择函数的名称,如['normGeomSelect']

selectOps-------传递个选择函数的参数,如[0.08]

xOverFNs-------交叉函数名称表，以空格分开，如['arithXover heuristicXover simpleXover']

xOverOps-------传递给交叉函数的参数表，如[2 0;2 3;2 0]

mutFNs-------变异函数表，如['boundaryMutation multiNonUnifMutation nonUnifMutation unifMutation']

mutOps-------传递给交叉函数的参数表,如[4 0 0;6 100 3;4 100 3;4 0 0]

GA工具箱

工具箱函数可以通过图形界面或Matlab命令行来访问，它们是用Matlab语言编写的，对用户开放，因此可以查看算法，修改源代码或生成用户函数。

遗传算法与直接搜索工具箱有助于求解那些不易用传统方法解决的问题，譬如旅行商问题等。

遗传算法与直接搜索工具箱有一个精心设计的用户图形界面，可以直观、方便、快速地求解最优化问题。

GA工具箱

1．功能特点

遗传算法与直接搜索工具箱的功能特点如下

（1）用户图形界面和命令行函数可用来快速地描述问题、设置算法选项以及监控进程。

（2）具有多个选项的遗传算法工具可用于问题创建、适应度计算、选择、交叉和变异。

GA工具箱

（3）直接搜索工具实现了一种模式搜索方法，其选项可用于定义网格尺寸、表决方法和搜索方法。

（4）遗传算法与直接搜索工具箱函数可与 Matlab 的优化工具箱或其它的 Matlab 程序结合使用。

（5）支持自动的 M 代码生成。

GA工具箱

1．在命令行使用遗传算法，可以用下列语法调用遗传算法函数 ga

[x, fval]=

ga(@fitnessfun,nvars,A,b,Aeq,beq,LB,UB,@nonlcon,options)

其中@fitnessfun 是目标函数句柄，nvars 是目标函数中独立变量的个数，options 是一个包含遗传算法选项参数的数据结构，其它参数的含义与非线性规划 fmincon 中的参数相同。函数返回值 x 为最终值到达的点，这里 x 为行向量，fval 为目标函数的最终值。

GA工具箱

求下列问题的解

$$\max\ f(x)=2x_1+3x_1^2+3x_2+x_2^2+x_3,$$

$$\text{s.t.}\begin{cases} x_1+2x_1^2+x_2+2x_2^2+x_3\le 10,\\ x_1+x_1^2+x_2+x_2^2-x_3\le 50,\\ 2x_1+x_1^2+2x_2+x_3\le 40,\\ x_1^2+x_3=2,\\ x_1+2x_2\ge 1,\\ x_1\ge 0,\quad x_2,x_3\text{不约束.} \end{cases}$$

GA工具箱

解　（1）编写适应度函数（文件名为 ycfun1.m）

```
function y=ycfun1(x);      %x 为行向量
c1=[2 3 1]; c2=[3 1 0];
y= c1* x' + c2* x'.^2; y=-y;
```

（2）编写非线性约束函数（文件名为 ycfun2.m）

```
function [f,g]=ycfun2(x);
f=[x(1)+2*x(1)^2+x(2)+2*x(2)^2+x(3)-10
    x(1)+x(1)^2+x(2)+x(2)^2-x(3)-50
    2*x(1)+x(1)^2+2*x(2)+x(3)-40];
g=x(1)^2+x(3)-2;
```

（3）主函数

```
clc, clear
a=[-1 -2 0;-1 0 0];b=[-1;0];
[x,y]=ga(@ycfun1,3,a,b,[],[],[],[],@ycfun2);
x, y=-y
```

GA工具箱

遗传算法程序的运行结果每一次都是不一样的，要运行多次，找一个最好的结果。

Part 4　粒子群算法

（Particle Swarm Optimizer, PSO）

基于群智能方法的演化计算技术

粒子群优化算法

- 粒子群优化算法(PSO)最初是由Kennedy和Eberhart博士于1995年受人工生命研究的结果启发，在模拟鸟群觅食过程中的迁徙和群集行为时提出的一种基于群体智能的演化计算技术。该算法具有并行处理、鲁棒性好等特点，能以较大概率找到问题的全局最优解，且计算效率比传统随机方法高。其最大的优势在于编程简单，易实现、收敛速度快，而且有深刻的智能背景，既适合科学研究，又适合工程应用。因此，PSO一经提出，立刻引起了演化计算领域研究者的广泛关注，并在短短几年时间里涌现出大量的研究成果，该算法目前已被“国际演化计算会议”列为讨论专题之一。
- PSO是受到鸟群或者鱼群社会行为的启发而形成的一种基于种群的随机优化技术。它是一类随机全局优化技术，通过粒子间的相互作用发现复杂搜索空间中的最优区域。该算法是一种基于群体智能的新型演化计算技术，具有简单易实现、设置参数少、全局优化能力强等优点。粒子群优化算法已在函数优化、神经网络设计、分类、模式识别、信号处理、机器人技术等许多领域取得了成功的应用。

粒子群优化算法

- 产生背景
- 设想这样一个场景：一群鸟随机的分布在一个区域中，在这个区域里只有一块食物。所有的鸟都不知道食物在哪里。但是他们知道当前的位置离食物还有多远。那么找到食物的最优策略是什么呢。最简单有效的方法就是追寻自己视野中目前离食物最近的鸟。如果把食物当作最优点，而把鸟离食物的距离当作函数的适应度，那么鸟寻觅食物的过程就可以当作一个函数寻优的过程。由此受到启发，经过简化提出了粒子群优化算法。

粒子群优化算法

- PSO的缺点：

  对于有多个局部极值点的函数，容易陷入到局部极值点中，得不到正确的结果。此外，由于缺乏精密搜索方法的配合，PSO方法往往不能得到精确的结果。再则，PSO方法提供了全局搜索的可能，但并不能严格证明它在全局最优点上的收敛性。因此，PSO一般适用于一类高维的、存在多个局部极值点而并不需要得到很高精度的优化问题。

粒子群优化算法

- **PSO算法的基本思想**

  每个优化问题的潜在解都是搜索空间中的一只鸟，称之为“粒子”。所有的粒子都有一个由被优化的函数决定的适应值（**fitness value**），每个粒子还有一个速度决定他们飞翔的方向和距离。然后粒子们就追随当前的最优粒子在解空间中搜索。**PSO**初始化为一群随机粒子（随机解）。然后通过迭代找到最优解。在每一次迭代中，粒子通过跟踪两个“极值”来更新自己。第一个就是粒子本身所找到的最优解。这个解称为个体极值。另一个极值是整个种群目前找到的最优解。这个极值是全局极值。另外也可以不用整个种群而只是用其中一部分作为粒子的邻居，那么在所有邻居中的极值就是局部极值。

粒子群优化算法

- PSO算法是一种启发式的优化计算方法，其最大的优点：
- (1)易于描述，易于理解；
- (2)对优化问题定义的连续性无特殊要求；
- (3)只有非常少的参数需要调整；
- (4)算法实现简单，速度快；
- (5)相对其它演化算法而言，只需要较小的演化群体；
- (6)算法易于收敛，相比其它演化算法，只需要较少的评价函数计算次　数就可达到收敛；
- (7)无集中控制约束，不会因个体的故障影响整个问题的求解，确保了系统具备很强的鲁棒性。

粒子群优化算法

•基本模型

设群体规模为N，在一个D维的目标搜索空间中，群体中的第i(i=1，2，…N)个粒子位置可以表示为一个D维矢量

$$X_i = (x_{i1}, x_{i2}, \cdots, x_{iD})^T$$

同时用 $V_i = (v_{i1}, v_{i2}, \cdots, v_{iD})^T, i = 1, 2, \cdots, N$

表示第i个粒子的飞翔速度。

用 $P_i = (p_{i1}, p_{i2}, \cdots, p_{iD})^T$ 表示第i个粒子自身搜索到的最好点。

而在这个种群中，至少有一个粒子是最好的，将其编号记为g，则 $P_g = (p_{g1}, p_{g2}, \cdots, p_{gD})^T$ 就是当前种群所搜索到的最好点，即种群的全局历史最优位置。

粒子群优化算法

- 粒子根据以下公式来更新其速度和位置：

$$v_{ij}^{k+1} = v_{ij}^{k} + c_1 r_{1j}(p_{ij}^{k} - x_{ij}^{k}) + c_2 r_{2j}(p_{gj}^{k} - x_{ij}^{k})$$

$$x_{ij}^{k+1} = x_{ij}^{k} + v_{ij}^{k+1}$$

其中i=1，2，…，N，j表示粒子的第j维，k表示迭代次数，c_1, c_2 为加速常数，一般在0~2之间取值。 主要是为了调节粒子自身的最好位置飞行的步长， v_{ij} 是为了调节粒子向全局最好位置飞行的步长。$r_1 \sim u(0,1)$ ， $r_2 \sim u(0,1)$ 为两个相互独立的随机函数。为了减少在进化过程中，粒子离开搜索空间的可能性，通常限定于一定范围内，即 $v_{ij} \in [V_{\min}, V_{\max}]$

粒子群优化算法

- 式(1)中其第一部分 v_{ij}^k 为粒子先前的速度；
- 其第二部分 $c_1 r_{1j}(p_{ij}^k - x_{ij}^k)$ 为“认知”部分，它仅考虑了粒子自身的经验，表示粒子本身的思考，
- 其第三部分 $c_2 r_{2j}(p_{gj}^k - x_{ij}^k)$ 为“社会”部分，表示粒子间的社会信息共享

粒子群优化算法

- 基本粒子群算法的流程如下：

(1)依照初始化过程，对粒子群的随机位置和速度进行初始设定；

(2)计算每个粒子的适应值；

(3)对于每个粒子，将其适应值与所经历过的最好位置 P_i 的适应值进行比较，若较好，则将其作为当前最好位置；

(4)对于每个粒子，将其适应值与全局所经历过的最好位置 P_g 的适应值进行比较，若较好，则将其作为当前的全局最好位置；

(5)根据两个迭代公式对粒子的速度和位置进行进化；

(6)如未达到结束条件通常为足够好的适应值或达到一个预设最大代数(Gmax)，返回步骤(2)；否则执行步骤(7)

(7)输出gbest.

粒子群优化算法

- 带惯性权重的粒子群算法

  为了改变基本粒子群算法的收敛性能，Y.Shi与R.C.Eberhart在1998年的IEEE国际进化计算学术会议上发表了题为“A Modified Particle Swarm Optimization”的论文。首先在速度进化方程中引入惯性权重(inertia weight) w，即

$$v_{ij}^{k+1} = wv_{ij}^{k} + c_1 r_{1j}(p_{ij}^{k} - x_{ij}^{k}) + c_2 r_{2j}(p_{gj}^{k} - x_{ij}^{k})$$

$$x_{ij}^{k+1} = x_{ij}^{k} + v_{ij}^{k+1}$$

粒子群优化算法

基本粒子群算法是w=1的特殊情况。把带惯性权重的微粒群算法称之为标准粒子群算法。由基本粒子群算法模型中粒子位置的进化方程可以看出，粒子在不同时刻的位置主要是由飞行速度决定的，也就是说粒子的飞行速度相当于搜索步长：

飞行速度的大小直接影响着算法的全局收敛性。当粒子的飞行速度过大时，各粒子初始将会以较快的速度飞向全局最优解邻近的区域，但是当逼近最优解时，由于粒子的飞行速度缺乏有效的控制与约束，则将很容易飞越最优解，转而去搜索其它区域，从而使算法很难收敛于最优解，陷入局部最优解；当粒子的飞行速度过小时，粒子在初期向全局最优解邻近区域靠近的搜索时间就需要很长。收敛速度慢，很难达到最优解。基于此现实情况，他们二人提出了标准的微粒群算法。

粒子群优化算法

- 式中的w为惯性权重，它具有维护全局和局部搜索能力的平衡作用，可以使粒子保持惯性运动，使其有扩展搜索空间的趋势，有能力探索新的区域。对全局搜索，通常的好方法是在前期有较高的搜索能力以得到合适的粒子，而在后期有较高的开发能力以加快收敛速度。为此，可将w设定为随着进化而线性减少，例如由0.9 ～1.2等。有些学者在研究中曾论证出w的最佳值在0.8附近，这将为设计标准微粒粒子群算法参数时提供了有利的参考。一般人们认为较大的w提高了寻优时粒子的全局搜索能力，有利于提高寻优的成功率；较小的w则有利于粒子群在迭代运算时的快速聚集，有利于提高寻优的速度。

Thank you!

Any question?