人工智能：模拟智能活动

- 人工智能研究的核心理念是“模拟智能活动的特征和结果”
- 模拟“单一”智能体
  - 自动推理、知识图谱、搜索技术等
- 模拟群体智能
  - 从自然界中适者生存的客观规律中获得启发
  - 物种在时间和空间上通过调整自身行为、属性来适应环境，使之生存并不断发展、延续

阿兰·图灵(1912-1954)

- 1950年，图灵提出，智能的本质是系统调整自身行为来适应不断变化的环境的能力，而与系统的形态、外观没有关系

模拟物种“适者生存”的能力，同样属于人工智能的研究范畴

群体智能是什么

- 群体智能（swarm intelligence）是指在集体层面表现出的分散的、去中心化的自组织行为
- 如蚁群、蜂群构成的复杂类社会系统，鸟群、鱼群为适应空气或海水而构成的群体迁移，以及微生物、植物在适应生存环境时候所表现的集体智能

蚁群能够搭建身体浮桥跨越缺口地形

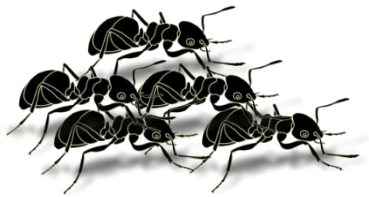
庞大的鱼群降低捕食者的成功机会

队形可以让鸟类减少15~20%的体力消耗

群体智能的基本原则

- 群体智能应该遵循五条基本原则（MillonasMM在1994年提出）：
  - 邻近原则，群体能够进行简单的空间和时间计算
  - 品质原则，群体能够响应环境中的品质因子
  - 多样性反应原则，群体的行动范围不应该太窄
  - 稳定性原则，群体不应在每次环境变化时都改变自身的行为
  - 适应性原则，在所需代价不太高的情况下，群体能够在适当的时候改变自身的行为

群体



群体必须在环境中表现出自主性、反应性、学习性和自适应性等智能特性

个体

在群体中个体并不需要相当复杂，反而可能很简单

群体智能的核心是由众多简单个体组成的群体能够通过相互之间的简单合作来完成某一任务

群体智能的特点

- 控制是分布式的，不存在中心控制
  - 即不会由于某一个或几个个体出现故障而影响群体对整个问题的求解
- 群体具有较好的可扩充性
  - 群体智能可以通过非直接通信的方式进行信息的传输与合作，因而随着个体数目的增加，通信开销的增幅较小
- 群体智能的实现方便，具有简单性
  - 群体中每个个体的能力或遵循的行为规则非常简单
- 群体具有自组织性
  - 群体表现出来的复杂行为是通过简单个体的交互过程实现出来的智能

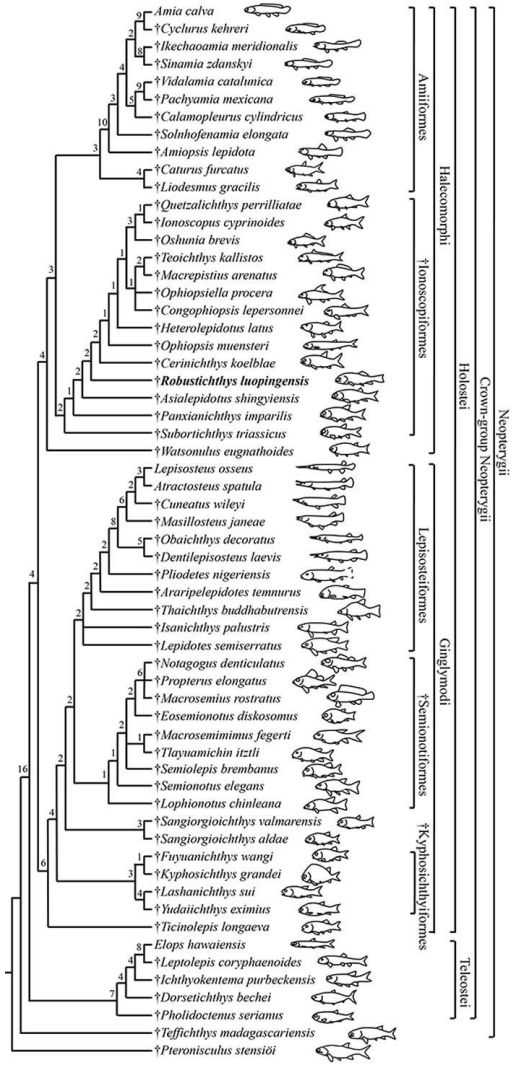
群体智能的两个维度

对物种适者生存能力的解读，可以分时间和空间两个维度来考虑

- **时间维度**
  - 种群通过代际繁衍，经历千万年的发展，不断进化以适应环境，借鉴物种随时间进化的过程来求解问题
- **空间维度**
  - 在同一时间，种群内的大量简单个体在限定条件下通过交互协作，使整体种群具有某种适应环境的能力，模拟这种群体行为产生的智能

空间维度

时间维度



全骨鱼类进化树

(Refer:http://www.cnfossil.com/viewnews-9142.html)

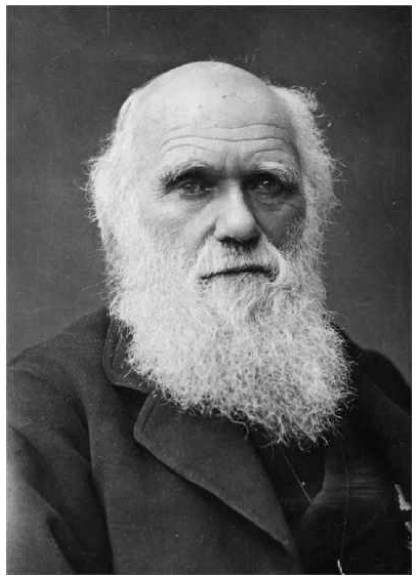
典型的群体智能算法

- 基本思想：模仿生物界的进化机理（时间维度）和群体协作（空间维度）行为

群体智能算法	基本思想
遗传算法	模拟生物种群进化
差分进化算法	模拟生物种群进化www
粒子群算法	模拟鸟类寻找食物
蚁群算法	模拟蚁群寻找食物
人工鱼群算法	模拟鱼类生活觅食的特性
萤火虫算法	模拟萤火虫在晚上的群聚活动
……	……

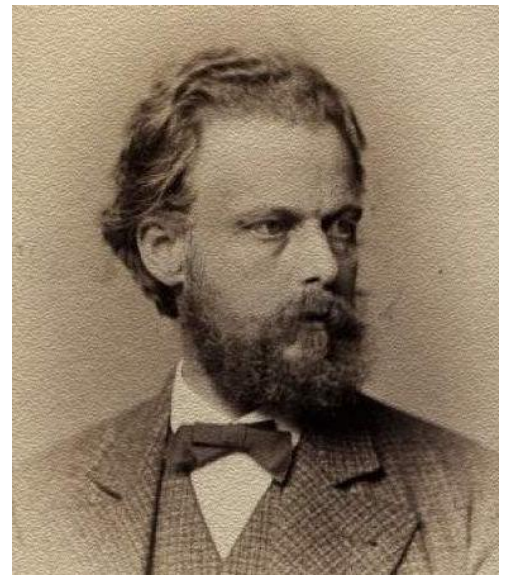
遗传算法的基本思想（Genetic Algorithm，GA)

- 模拟达尔文“优胜劣汰、适者生存”的原理
- 模拟孟德尔遗传变异理论
- 借鉴魏斯曼的自然选择理论



达尔文(1809-1882)

孟德尔(1822-1884)

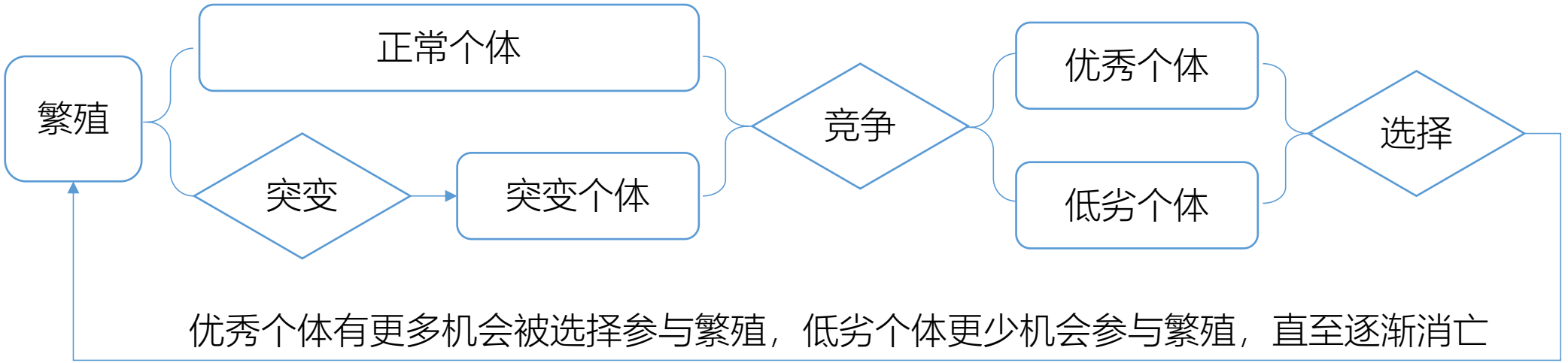


魏斯曼(1834-1914)

遗传算法的基本思想

- 生物进化包括四个过程：
  - 繁殖：物种代际延续的基础
  - 突变：物种适应不断变化环境的关键
  - 竞争、选择：物种优胜劣汰、不断优化的关键

长颈鹿祖先中，脖子短的吃不到食物逐渐被淘汰，若干代后所有的长颈鹿都是长颈的

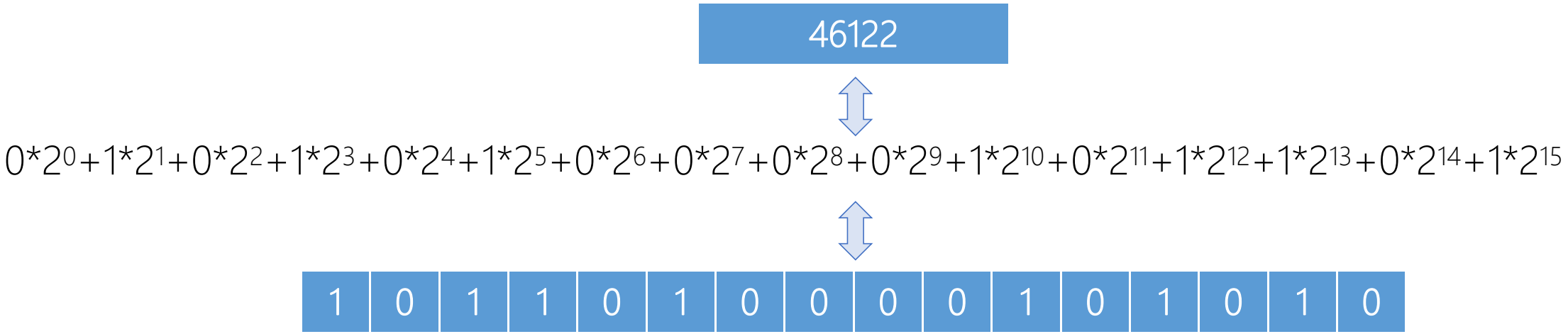


遗传算法的基本思想

- 进化智能算法的目标，就是用计算机来模拟进化过程，从而求解问题，最简单的进化算法，是由进化智能创始人之一的John Holland于1975年提出的遗传算法（Genetic Algorithm，GA），该算法在1989年经Goldberg汇总改进，形成完整的框架。
- 为了实现对进化论的模拟，遗传算法必须考虑以下几方面问题：
  ① 什么是物种？什么是物种的生存环境？
  ② 如何模拟物种的“繁殖”，产生下一代正常个体？
  ③ 如何模拟物种的“突变”， 产生下一代的突变个体？
  ④ 如何评价每个个体的优劣，从而模拟“竞争与选择”？

① 什么是物种

- 在遗传算法中，要求解的问题是“物种”，求解问题的每个可能的解，或者中间状态，都可以设计为该物种的“个体”
- 那物种如何表示呢？在现实世界中，区别生物个体性状由基因决定，个体的全部基因构成“染色体”
- 遗传算法借鉴了基因的思想：用符号串表示基因，若干基因构成染色体表示个体
- 最简单的染色体就是一串二进制数，其中的每个“位”就是基因



① 什么是物种的生存环境

- 环境设计包括：
  - ① 规定物种生存的限制条件
  - ② 规定物种适应环境的能力的指标，一般是量化指标
- 环境设计要保证最优解对应的个体具有最强的适应能力，最优解个体的“基因”有更高的概率在代际传递和保持

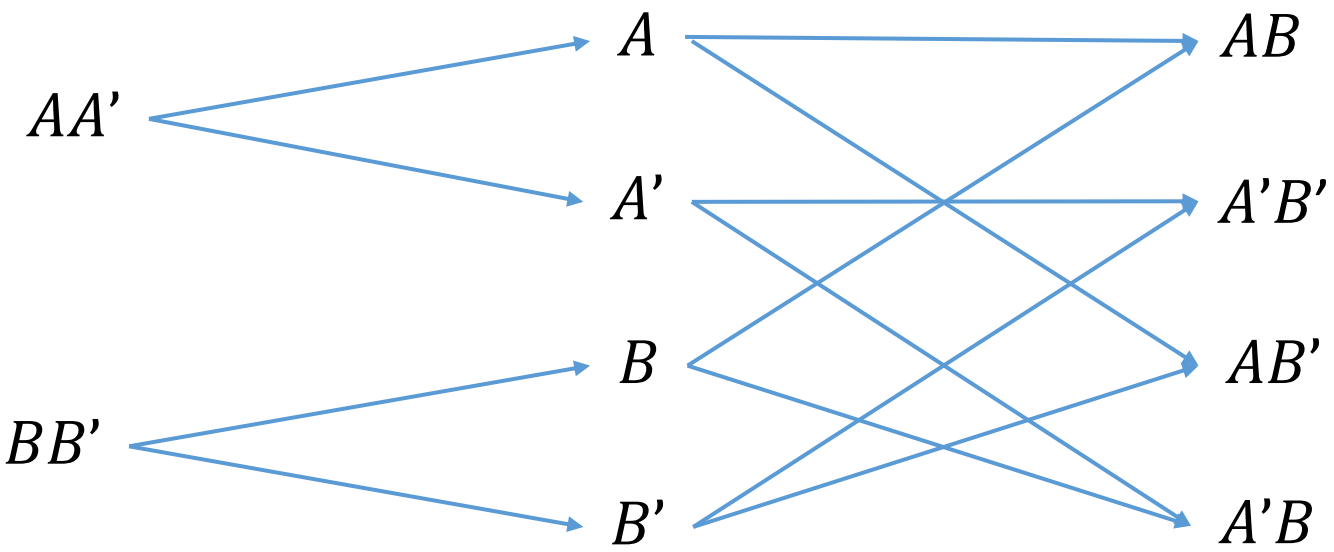
② 规定物种适应环境的能力的指标，称为适应度函数

① 物种生存的限制条件，即物种只能在给定的区域内生存繁衍

- 求函数$f(x) = 15x - x^2$在x在0~15时的最大值，其中x为整数

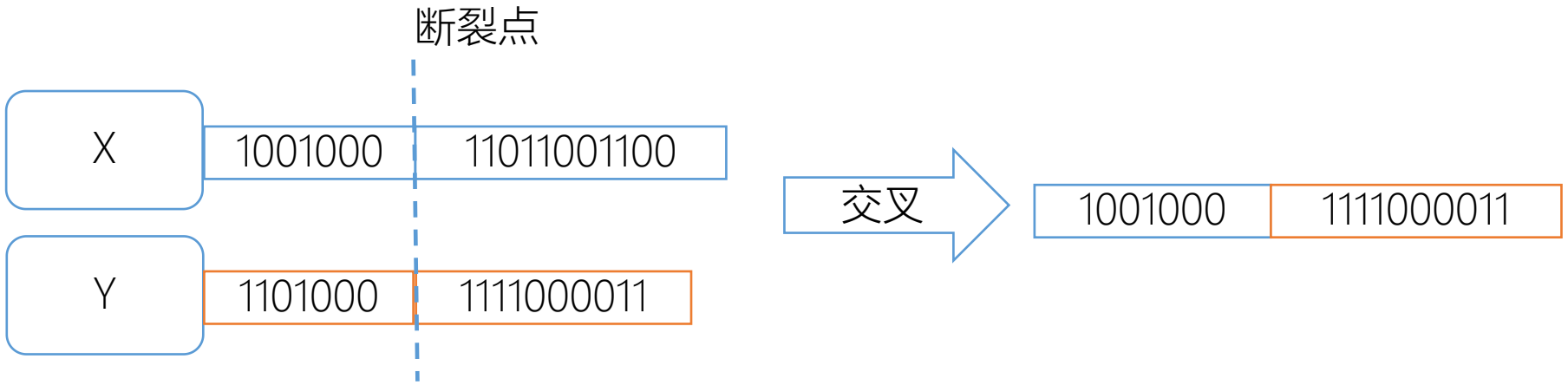
② 如何模拟物种的“繁殖”

- 从“基因”传承的角度来看繁殖过程
- 假设有两个个体X和Y结合产生后代，则繁殖的过程就是：X和Y分别随机拿出一半基因，组合形成新个体的基因
- 如X的基因为AA'，Y的基因为BB'，则后代的基因组合可能是：AB，AB'，$A'B$，$A'B'$四种之一
- 这就是“基因交叉”原则



② 如何模拟物种的“繁殖”

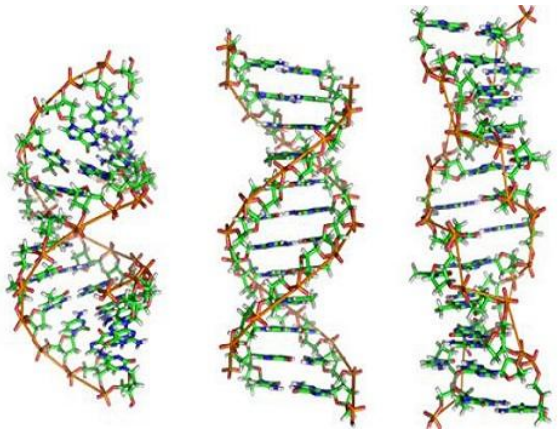
- 对于遗传算法中的繁殖，也可以通过基因交叉来实现
  - 从种群中选择两个个体
  - 随机确定一个染色体的“断裂点”，注意不一定是均分
  - 按照一定概率，交叉双亲的染色体生成新的个体



- 交叉基因的概率，称为遗传算法中的“交叉算子”

③ 如何模拟基因“突变”

- 繁殖过程的“基因交叉”实现了生物基因信息的“传递”，通过繁殖，优秀的基因可以在种群中不断复制、代代相传
- 但基因交叉不会给物种基因库增加新的“信息”，也就是说，如果环境发生变化，现有基因无法适应，则种群理论上就无法适应这种环境了
- 在现实中，生物基因存在小概率的“突变”，使得整个种群的基因能够跳跃性变化，从而从大量变化中产生能够适应新环境的基因

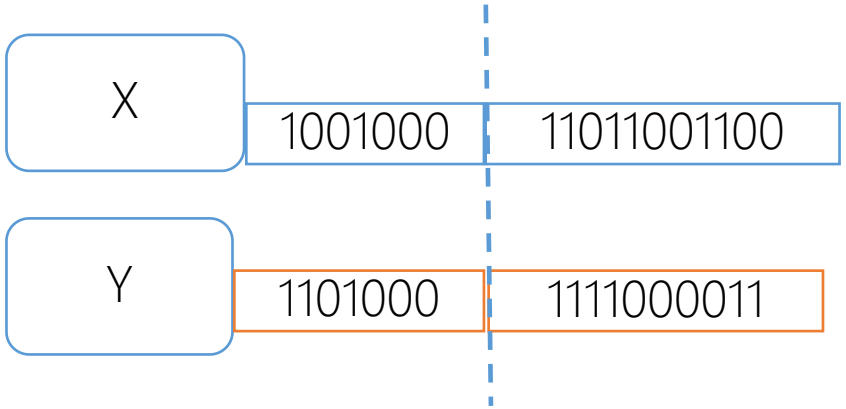


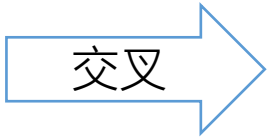
- CCR5基因的突变导致艾滋病痊愈

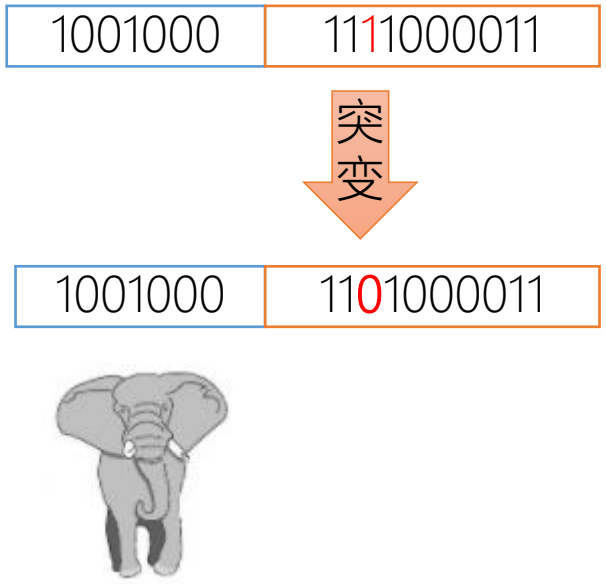
- 镰刀型细胞贫血症

③ 如何模拟基因“突变”

- 遗产算法同样模拟基因的突变过程
- 在实现基因交叉之后，产生的新的个体中的每个基因，都以小概率发生“突变”，从而产生具有全新基因的后代







- 基因发生突变的概率
- 称为遗传算法的“**突变算子**”

④ 如何模拟“竞争与选择”

- 竞争：
  - 遗传算法遵循自然界中的优胜劣汰原则，这就需要评价每个个体的适应能力，一般通过设计“适应度”函数来实现，适应度高的个体，更接近问题的最优解
- 选择：
  - 选择就是挑选优秀个体参与繁殖，产生下一代的过程，因此个体被选择的概率，与其适应度成正比
  - 个体被选择参与繁殖的概率，称为遗传算法的“**选择算子**”

④ 如何模拟“竞争与选择”

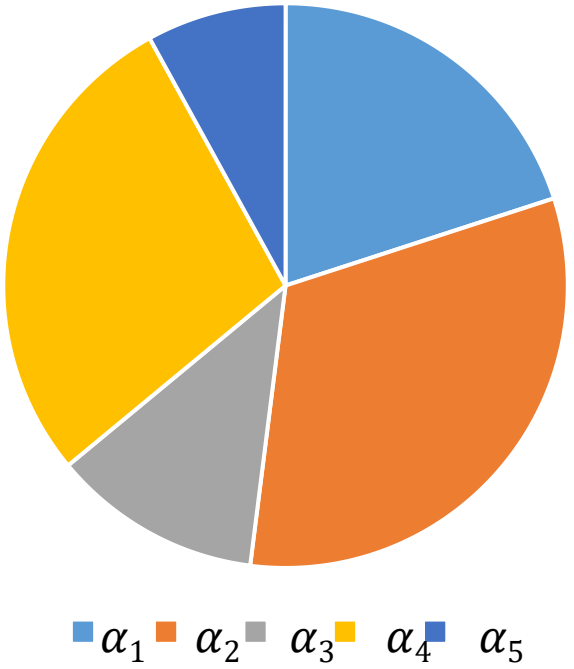
- 轮盘赌选择（roulette wheel selection）：是一种回放式随机采样方法。每个个体进入下一代的概率等于它的适应度值与整个种群中个体适应度值和的比例

可以看出，适应性越高的个体被选中的概率就越大

- 假如有 5 条染色体，适应度分别为 5 、 8 、 3 、 7 、 2
- 那么总的适应度为：$F = 5 + 8 + 3 + 7 + 2 = 25$
- 那么各个个体的被选中的概率为：

物竞天择，适者生存

$$\alpha_1 = (5/25) * 100\% = 20\%$$
$$\alpha_2 = (8/25) * 100\% = 32\%$$
$$\alpha_3 = (3/25) * 100\% = 12\%$$
$$\alpha_4 = (7/25) * 100\% = 28\%$$
$$\alpha_5 = (2/25) * 100\% = 8\%$$



遗传算法的流程

- 算法初始设定阶段:
  - Step1：根据求解问题，设计个体染色体表示方式，一般为二进制数字串
  - Step2：随机产生N个个体，作为初始种群
  - Step3：设定交叉概率Pc，突变概率为Pm，选择概率为Ps
  - Step4：设定个体的适应度函数$f(x)$，计算初始种群中的每个个体的适应度
- 进化阶段:
  - Step5：在当前种群中，根据Ps概率（可用轮盘赌选择得到），选择一对染色体作为"双亲"参与繁殖
  - Step6：根据Pc和Pm概率，对选中的双亲进行基因交叉、突变，生成后代染色体，加入下一代种群中
  - Step7：回到Step5重复繁殖过程，直到下一代种群数量达到N，用新种群（其中均为新一代）替换初始种群
- 评价阶段:
  - Step8：评价新的种群是否有个体满足停止条件，如达到要求，停止算法，返回最优个体。

    否则，回到Step5继续繁殖下一代。

遗传算法实例

- 求解问题：
  - 求函数$f(x) = 15x - x^2$ x在0~15时的最大值，其中x为整数
- Step1：设计染色体
  - x为0~15范围内的整数，取值一共有16种方式
  - 只需要4个二进制位就可以表示，因此染色体对应的就是$0000 \sim 1111$
- Step2~4：初始化
  - 初始种群数量$N = 6$，随机产生的0、1串，填充6个染色体对应的所有基因，创建初始种群
  - 交叉概率$Pc = 0.7$，即有70%可能在繁殖时引起交叉
  - 突变概率$Pm = 0.001$，即每个基因突变的概率
  - 选择概率Ps根据种群情况而定，每代不同
  - 适应度函数，即目标问题：$f(x) = 15x - x^2$

遗传算法实例

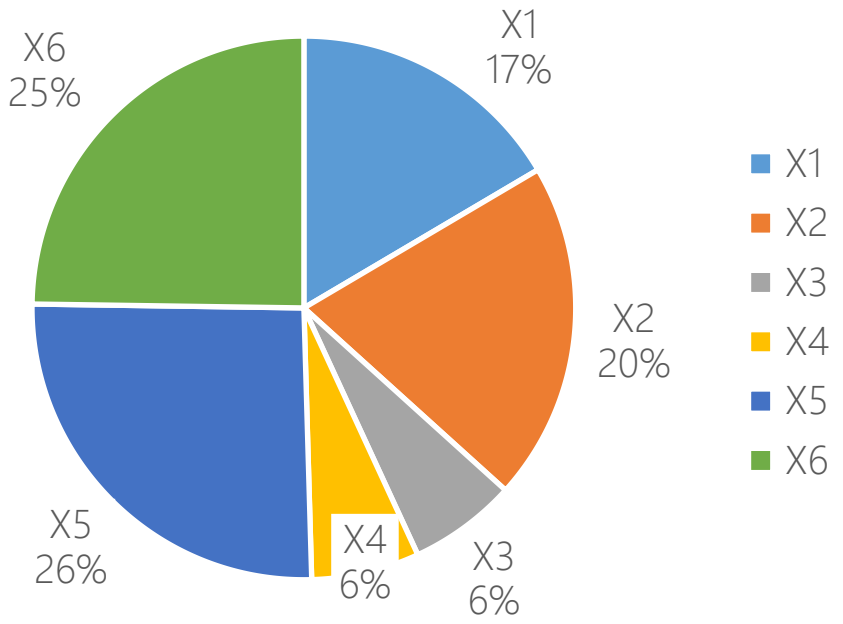
- Step2~4：初始化
  - 根据设定，随机初始化种群，并且计算个体适应度，如下表：

种群的平均适应度为（218/6）=36.33

染色体编号	染色体(随机产生)	X值	适应度 f(x)	选择概率Ps（%）
X1	1100	12	36	16.5
X2	0100	4	44	20.2
X3	0001	1	14	6.4
X4	1110	14	14	6.4
X5	0111	7	56	25.7
X6	1001	9	54	24.8

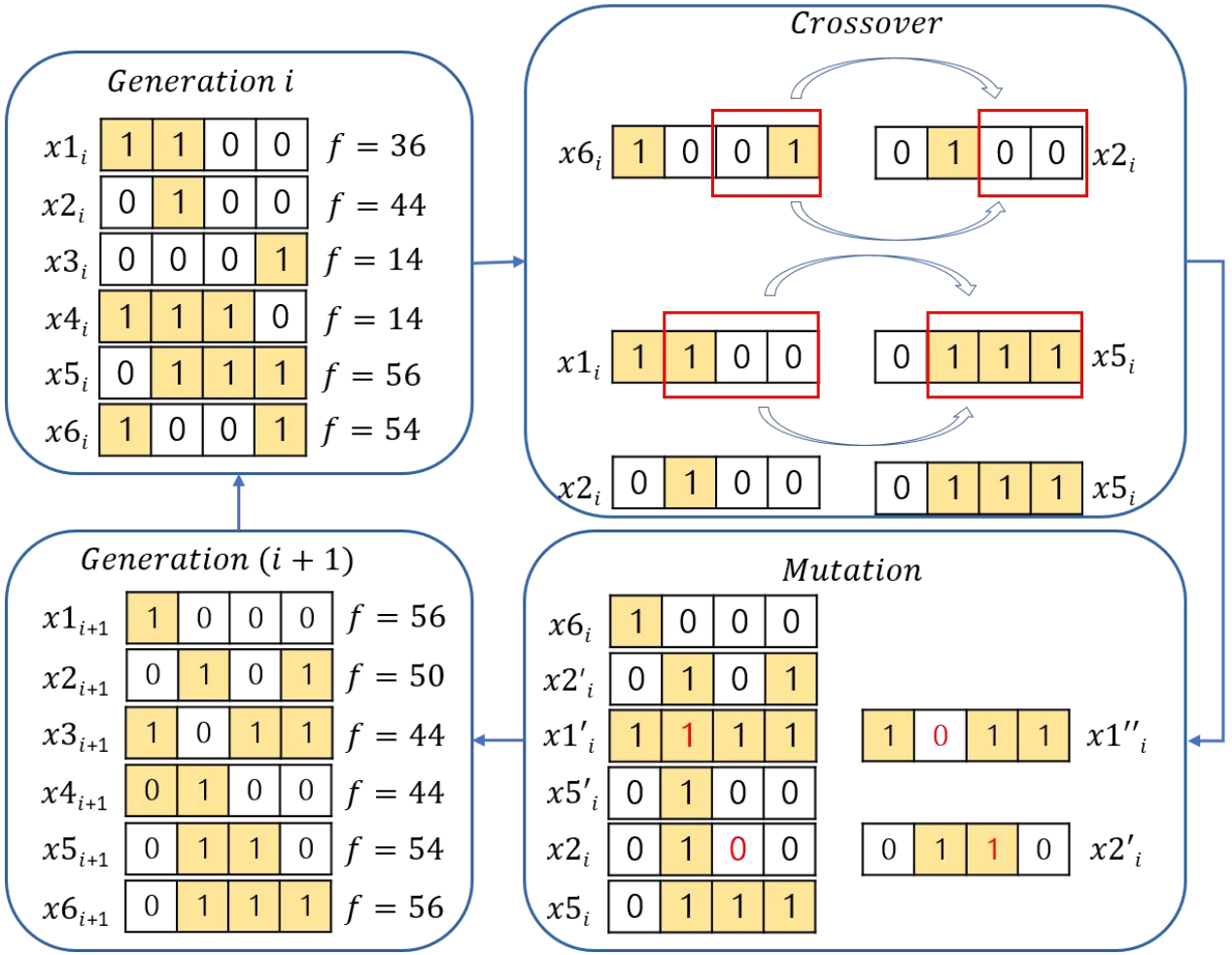
例如： 个体$X2$，其适应度为$f(4) = 15 \times 4 - 4^2 = 44$，则$Ps(X2) = 44 \div 218 = 0.202$

遗传算法实例



- Step5：选择双亲
  - 根据当前种群的情况，计算每个个体被选择的概率Ps
  - 将每个个体的适应度占种群适应度的权重作为概率
  - 计算所有个体的选择概率，得到选择轮盘，这就是“轮盘选择”：

遗传算法实例

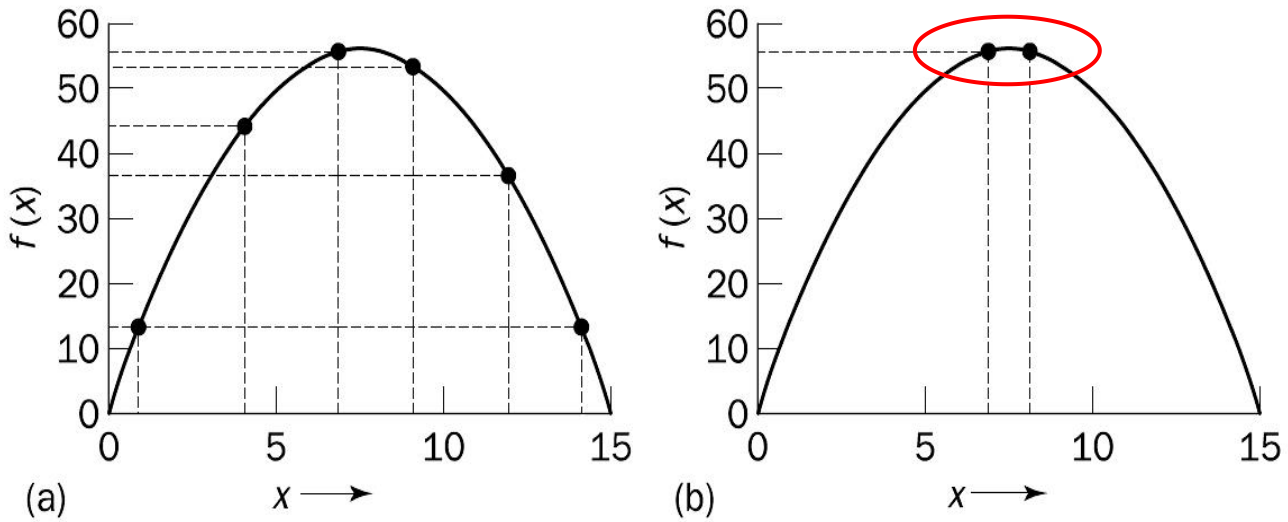


- Step6~7：交叉和突变
  - 本例中选择3对双亲，分别为：

    $X6-X2$； $X1-X5$； $X2-X5$；
  - 对每对染色体，进行交叉和突变
  - 按照$Pc=0.7$的概率，确定是否被选中进行基因交叉，断裂点的选择完全随机
  - 每对染色体在交叉之后，形成一对新的染色体
  - 按照$Pm=0.001$的概率，对染色体的基因位进行突变，得到最后的繁殖结果

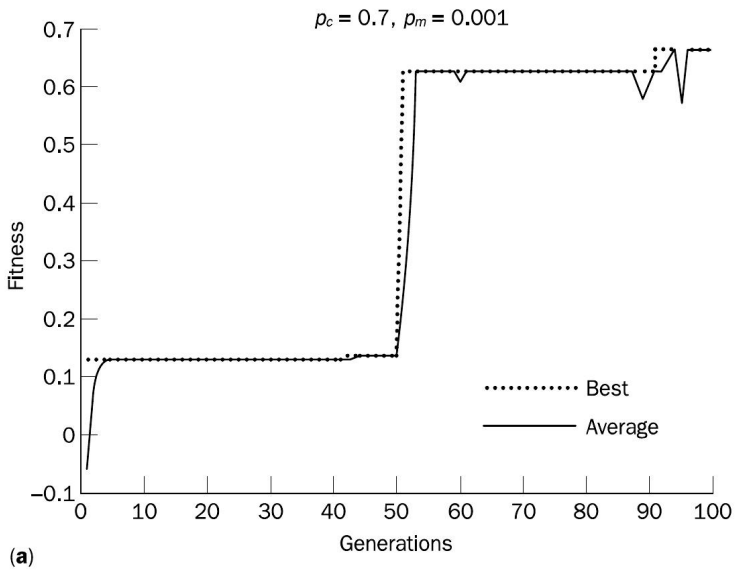
遗传算法实例

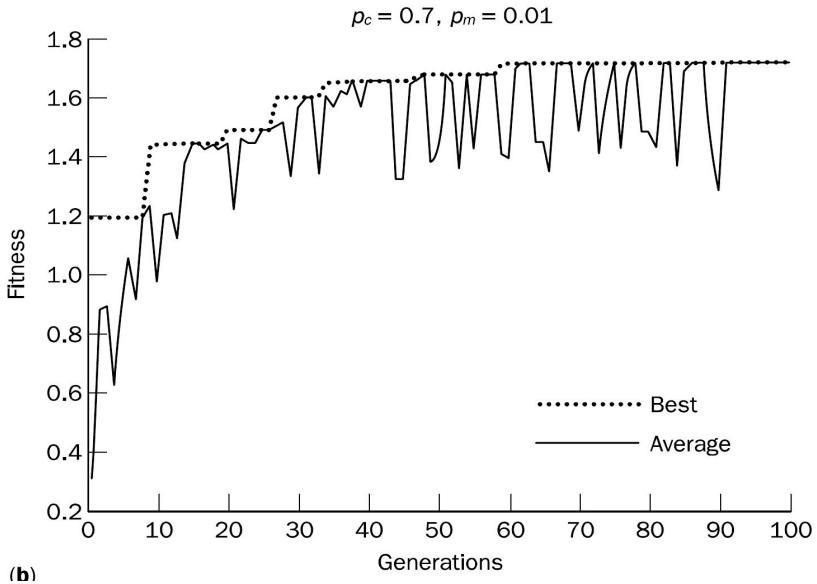
- 循环运行进化阶段
- 经过若干代繁殖，最终种群仅包含染色体0111和1000，代表了适应性最好的两类个体

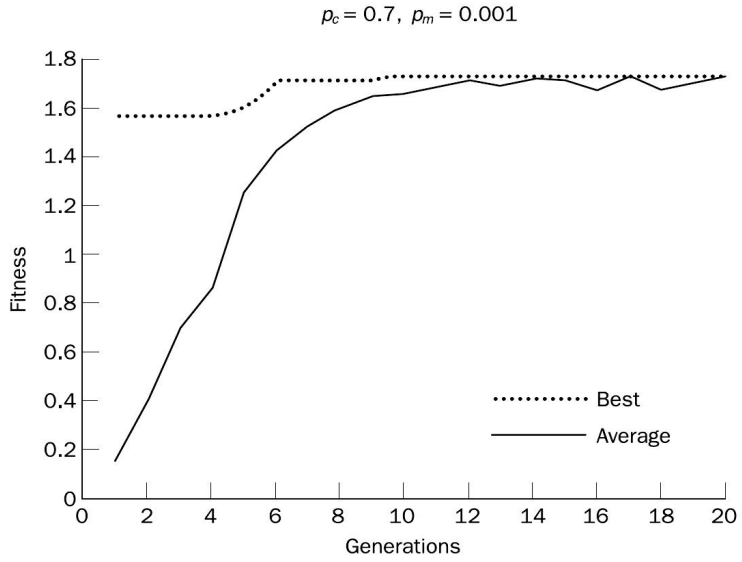


遗传算法实例

- 在本例中，可将每个世代的平均适应度画成图，以观察种群的适应度变化







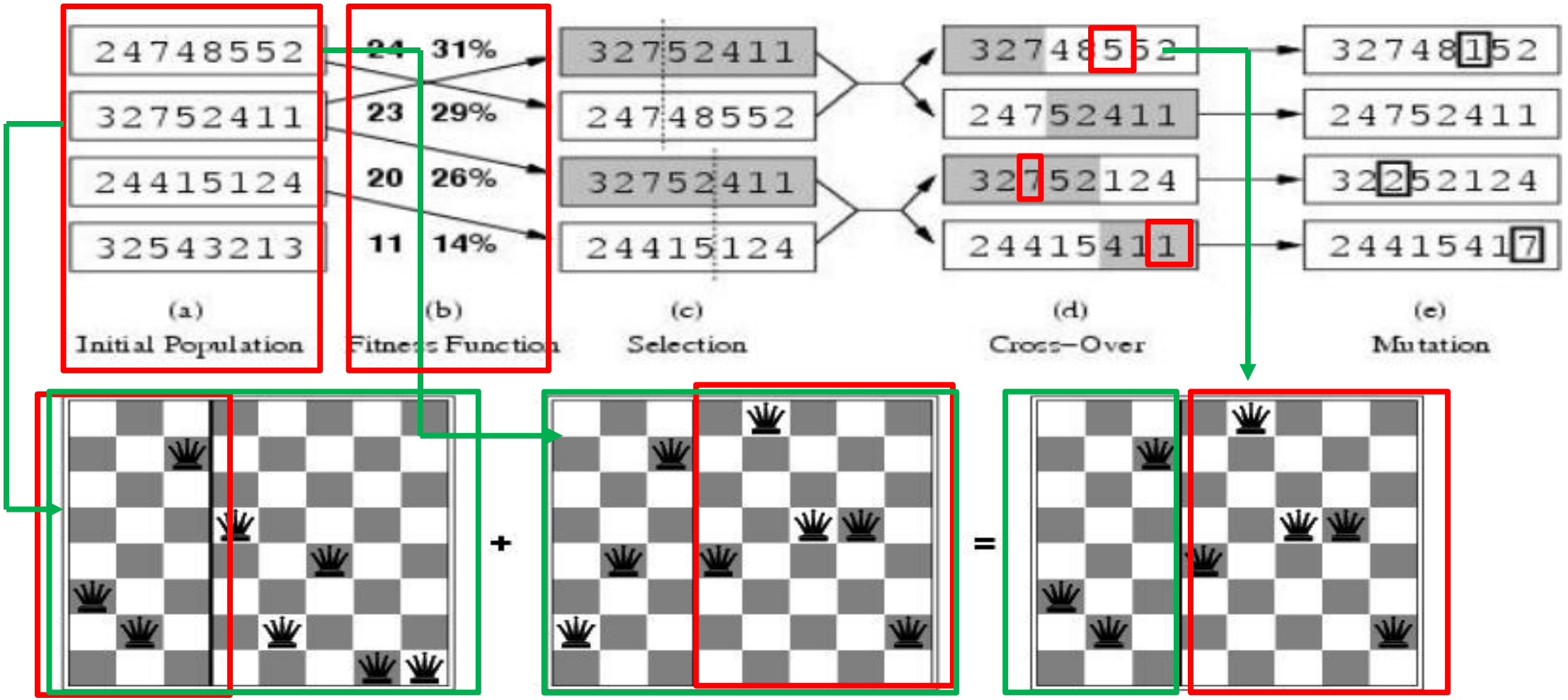
突变概率较小时，大多数情况下物种的适应性比较稳定，求解速度慢，但优秀的突变会在短时间内迅速提高整个种群的适应性

将突变概率扩大10倍，种群适应性波动明显增大，优秀个体在较早期就能够出现，但持续度不强，容易被再次突变抹杀

显然单纯提高突变概率，并不能解决问题，可将种群数量提高到60，则求解速度和稳定性都有明显提升

遗传算法求解N-皇后

- 8皇后问题：
- 适应值函数：相互攻击不到的皇后对数 (min = 0, max = (8×7)÷2 = 28)



遗传算法

```
function GENETIC-ALGORITHM(population, FITNESS-FN) returns an individual
  inputs: population, a set of individuals
          FITNESS-FN, a function that measures the fitness of an individual
  repeat
    new_population ← empty set
    for i = 1 to SIZE(population) do
      x ← RANDOM-SELECTION(population, FITNESS-FN)
      y ← RANDOM-SELECTION(population, FITNESS-FN)
      child ← REPRODUCE(x, y)
      if (small random probability) then child ← MUTATE(child)
      add child to new_population
    population ← new_population
  until some individual is fit enough, or enough time has elapsed
  return the best individual in population, according to FITNESS-FN
```

粒子群算法（Particle Swarm Optimization，PSO）

- 粒子群算法，也称粒子群优化算法或鸟群觅食算法（Particle Swarm Optimization，PSO），是一种基于种群的随机优化技术，由Kennedy和Eberhart于1995年提出

- 粒子群算法模仿昆虫、兽群、鸟群和鱼群等的群集行为，利用群体中的个体对信息的共享使整个群体的运动在问题求解空间中产生从无序到有序的演化过程，从而获得最优解

粒子群算法的基本思想

- 设想这样一个场景：
  - 一群鸟在随机搜索食物
  - 在这块区域里只有一块食物
  - 所有鸟都不知道食物在哪里
  - 每只鸟都知道当前时刻离食物最近的鸟的位置
  - 它们能感受到当前的位置离食物还有多远

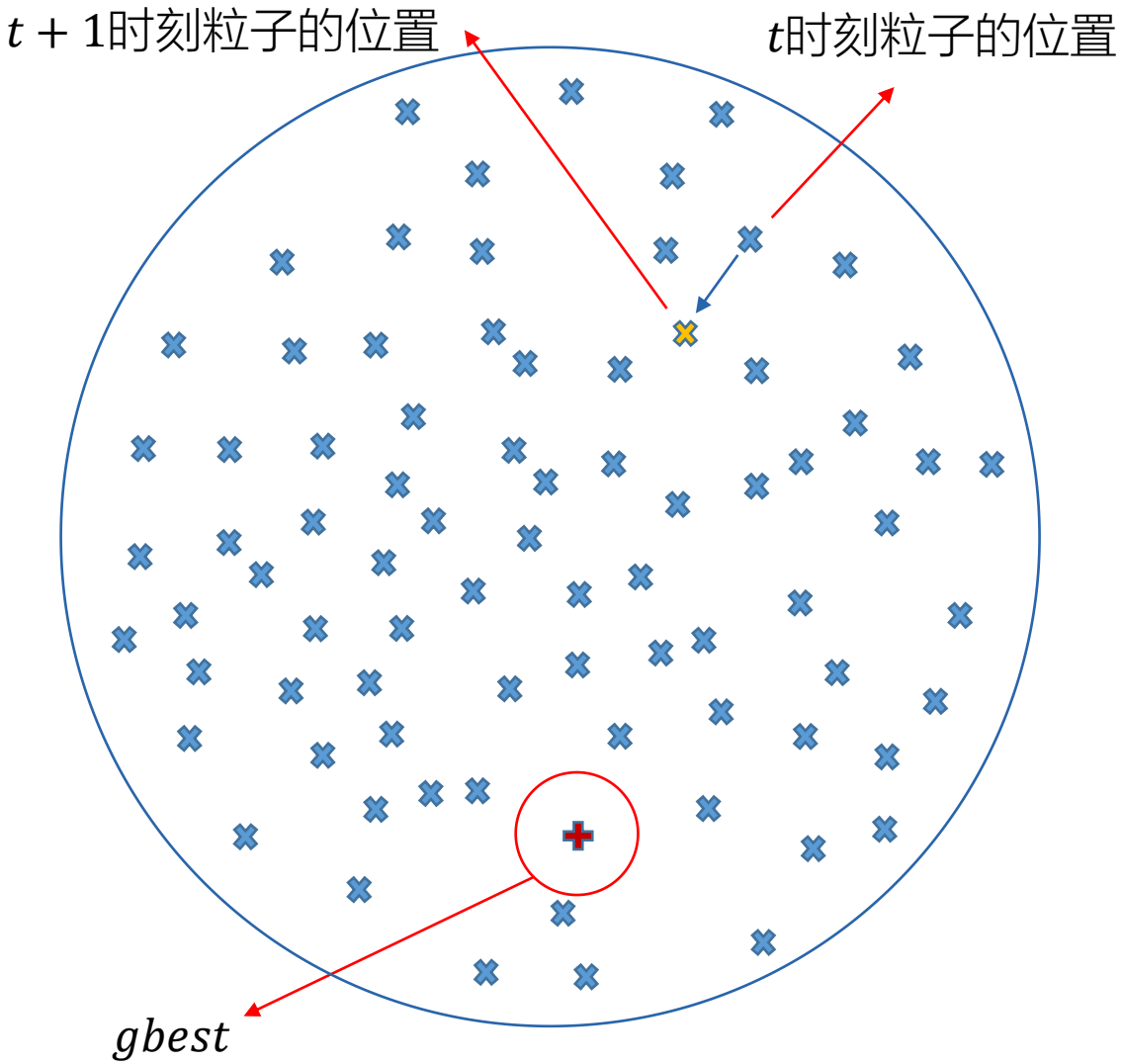
已知

- 那么找到食物的最优策略是什么呢？
  - 搜寻目前离食物最近的鸟的周围区域

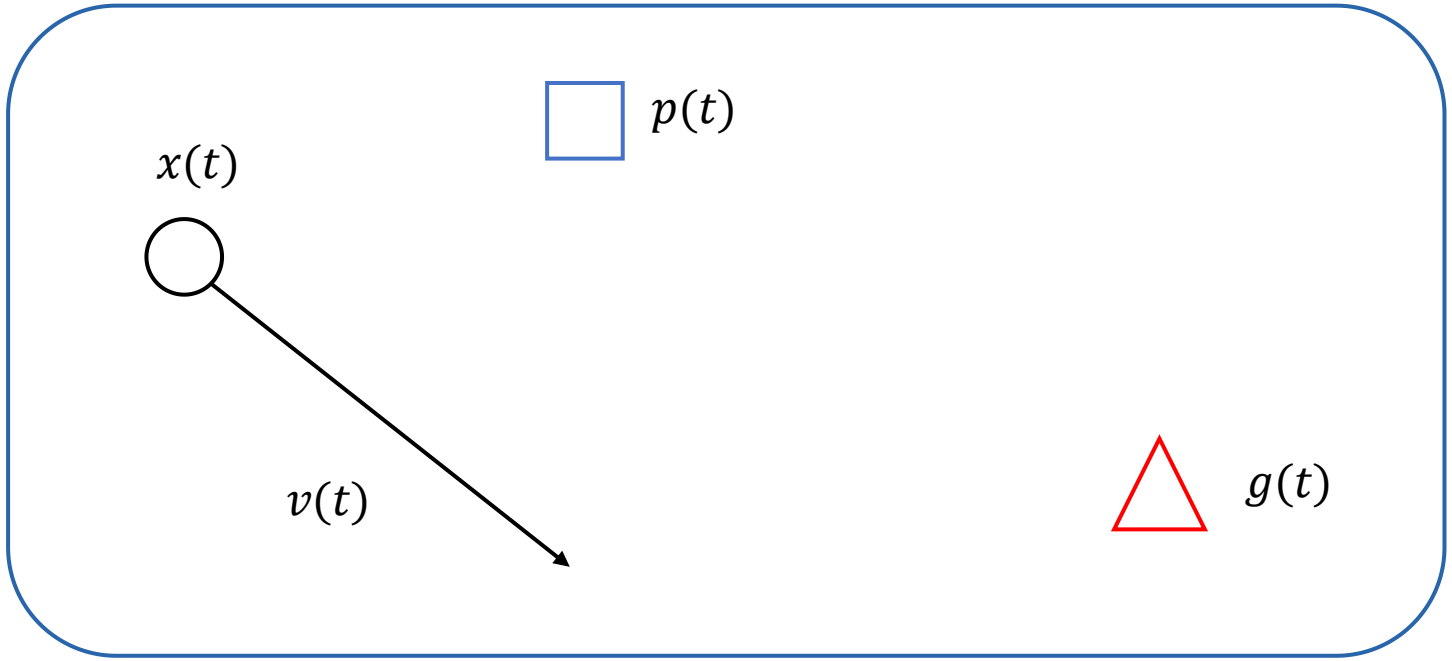
- 粒子群优化算法的基本思想是通过群体中个体之间的协作和信息共享来寻找最优解

问题抽象

- 鸟被抽象为没有质量和体积的粒子，并延伸到N 维空间
- 粒子在N 维空间的位置表示为向量$X_i = (x_1, x_2, \ldots, x_N)$，飞行速度表示为向量$V_i = (v_1, v_2, \ldots, v_N)$
- 粒子自己的状态
  - 每个粒子都有一个由目标函数决定的适应值，并且知道自己到目前为止发现的最好位置$pbest$和现在的位置X_i
- 粒子同伴的状态
  - 每个粒子还知道到目前为止整个群体中所有粒子发现的最好位置$gbest$（$gbest$是$pbest$中的最好值）
- 粒子通过自己的状态和同伴的状态来决定下一步的运动

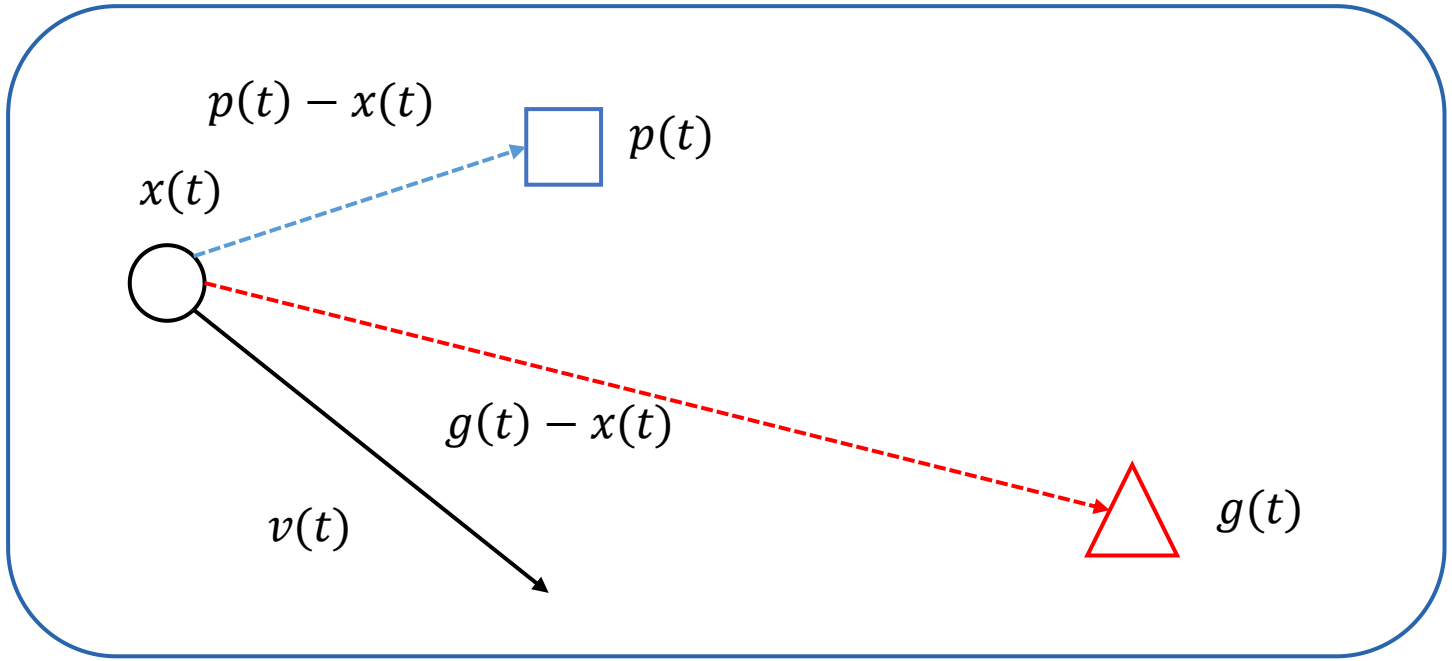


粒子位置与速度更新



- 将每个个体表示为粒子，每个个体在某一时刻的位置表示为$x(t)$，方向表示为$v(t)$
- $p(t)$为在t时刻x个体自己的最优解，$g(t)$为在t时刻群体的最优解

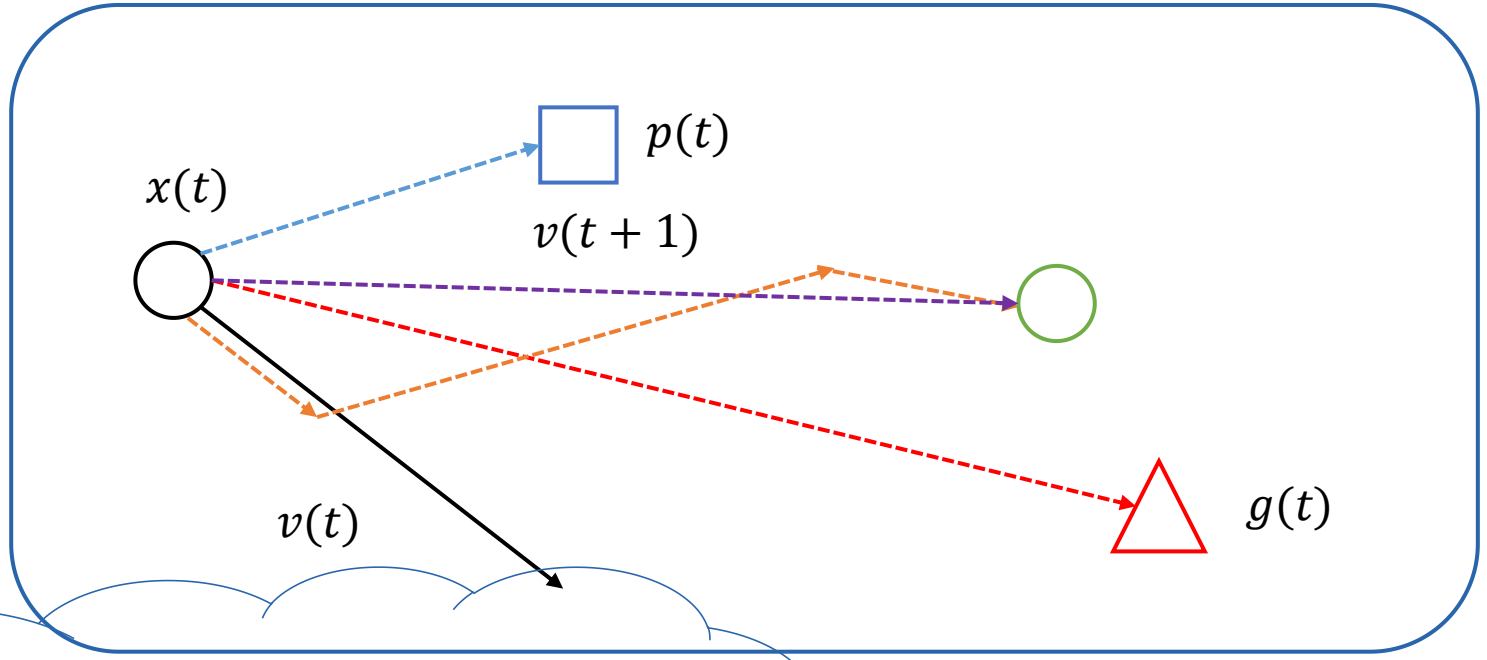
粒子位置与速度更新



- 下一个位置为左图所示由$x(t)$，$p(t)$，$g(t)$共同决定
- 下一个位置移动的方向受$p(t) - x(t)$，$g(t) - x(t)$，$v(t)$向量方向的影响
- 种群中的粒子通过不断地向自身和种群的历史信息进行学习，从而可以找到问题最优解

粒子位置与速度更新

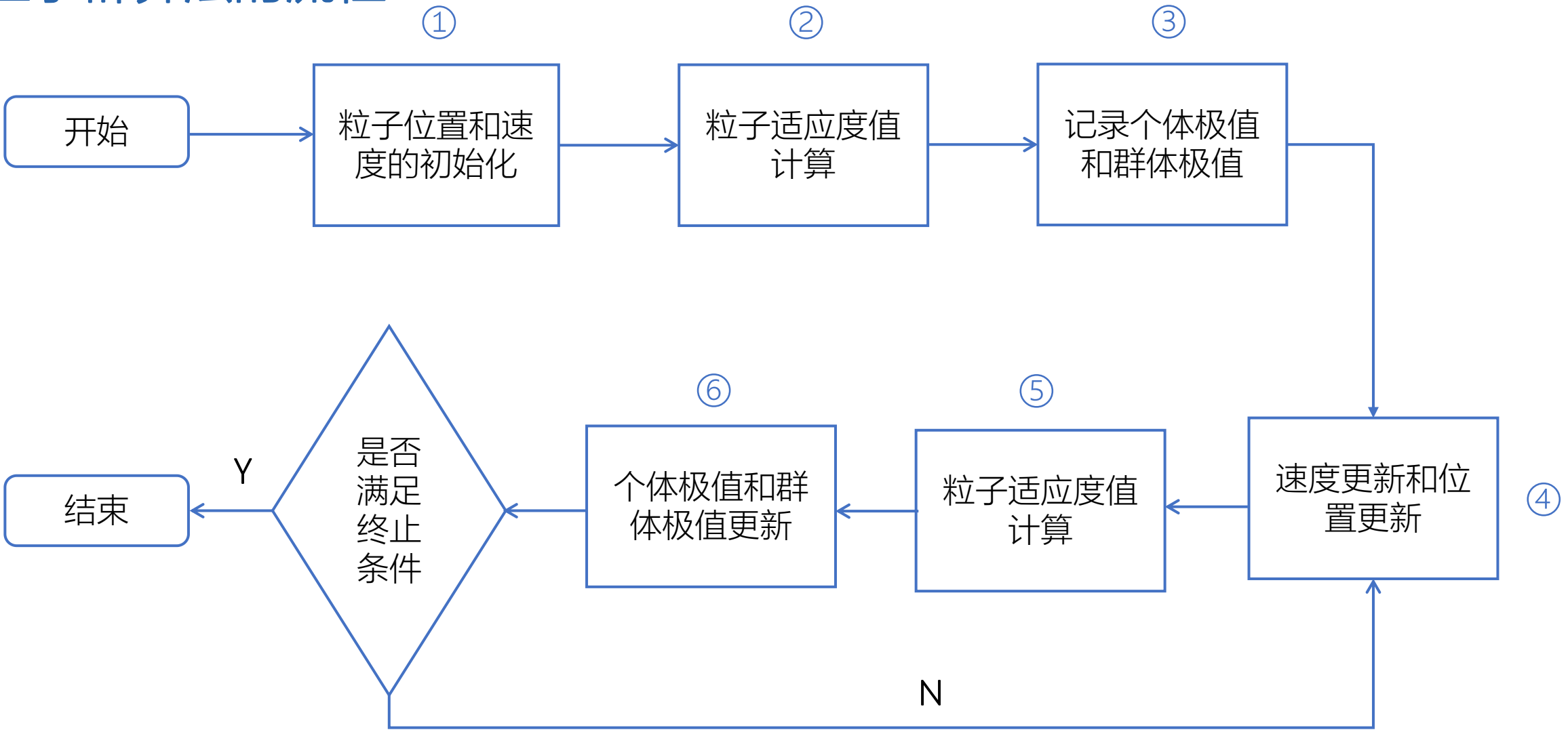
- 速度与位置更新公式
  - $v(t+1) = w * v(t) + c1 * rand() * \big(p(t) - x(t)\big) + \ c2 * rand() * (g(t) - x(t))$
  - $x(t+1) = x(t) + v(t+1)$



- w：惯性因子，其值为非负
  - 其值较大时，全局寻优能力强，局部寻优能力弱
  - 其值较小时，全局寻优能力弱，局部寻优能力强
- $rand()$：介于0-1之间的随机数
- $c1$和$c2$：学习因子

可设置动态惯性因子，在迭代前期使用较大值，提高全局寻优能力，在迭代后期使用较小值，提高局部寻优能力

粒子群算法的流程


①
②
③
开始
粒子位置和速度的初始化
粒子适应度值计算
记录个体极值和群体极值
速度更新和位置更新
④
⑤
粒子适应度值计算
⑥
个体极值和群体极值更新
是否满足终止条件
Y
结束
N

粒子群算法与遗传算法对比

- 粒子群算法相比于遗传算法更加的简单易懂，其调节参数较少
- 不同点
  - PSO由于不需要编码，没有GA算法中由于编码带来的求解精度限制
  - 在信息共享方面，大多数情况下所有粒子可能比遗传算法中的进化个体以更快速度收敛
    - GA算法，染色体之间共享信息，所以整个种群的移动是比较均匀地向最优区域移动
    - PSO算法，粒子通过当前搜索到最优点进行共享信息，很大程度上这是一种单项信息共享机制，整个搜索更新过程是跟随当前最优解的过程
- 相似点
  - 都为仿生算法，求解过程中存在多个解，因此都可以进行并行运算
  - 对个体的适配信息进行搜索，因此不受函数约束条件的限制，如连续性、可导性等

粒子群算法实例

- 求解问题：
  - 求函数y = $f(x_1, x_2) = {x_1}^2 - {x_2}^2$ 的最小值，其中$-10 \le x1, x2 \le 10$
- 步骤①-③：初始化
  - ① 初始种群数量$N = 3$，在搜索空间中随机初始化每个解的速度和位置
  - ② 计算适应度函数
  - ③ 得到粒子的历史最优位置和种群的全局最优位置

$$\mathrm{p1} = \begin{cases} v1 = (3,\ 2) \\ \mathrm{x1} = (8, -5) \end{cases} \quad \begin{cases} f1 = 8^2 - (-5)^2 = 39 \\ pBest1 = x1 = (8, -5) \end{cases}$$

$$\mathrm{p2} = \begin{cases} v2 = (-3, -2) \\ x2 = (-5,9) \end{cases} \quad \begin{cases} f2 = (-5)^2 - 9^2 = -56 \\ pBest2 = x2 = (-5,9) \end{cases}$$

$$\mathrm{p3} = \begin{cases} v3 = (5,3) \\ x3 = (-7, -8) \end{cases} \quad \begin{cases} f3 = (-7)^2 - (-8)^2 = -15 \\ pBest3 = x3 = (-7, -8) \end{cases}$$

$$\mathrm{gBest} = \mathrm{pBest2} = (-5,9)$$

粒子群算法实例

- 步骤④：粒子的速度和位置更新
  - 根据自身的历史最优位置和全局最优位置，更新每个粒子的速度和位置

$$p_1=\begin{cases} v_1 = w\times v_1 + c_1\times r_1\times(pBest_1 - x_1) + c_2\times r_2\times(gBest - x_1) \\ \Rightarrow v_1 = \begin{cases} 0.5\times 3 + 0 + 2\times 0.3\times(-5-8) = -6.3 \\ 0.5\times 2 + 0 + 2\times 0.1\times(9-(-5)) = 3.8 \end{cases} = (-6.3, 3.8) \\ x_1 = x_1 + v_1 = (8,-5) + (-6.3, 3.8) = (1.7, -1.2) \end{cases}$$

- w是惯性权重，这里假设为0.5
- c_1和c_2为加速系数，通常去固定值2.0
- r_1和r_2是[0,1]区间的随机数

$$p_2=\begin{cases} v_2 = w\times v_2 + c_1\times r_1\times(pBest_2 - x_2) + c_2\times r_2\times(gBest - x_2) \\ \Rightarrow v_2 = \begin{cases} 0.5\times(-3) + 0 + 0 = -1.5 \\ 0.5\times(-2) + 0 + 0 = -1.0 \end{cases} = (-1.5, -1.0) \\ x_2 = x_2 + v_2 = (-5,9) + (-1.5, -1.0) = (-6.5, 8) \end{cases}$$

$$p_3=\begin{cases} v_3 = w\times v_3 + c_1\times r_1\times(pBest_3 - x_3) + c_2\times r_2\times(gBest - x_3) \\ \Rightarrow v_3 = \begin{cases} 0.5\times 5 + 0 + 2\times 0.05\times(-5-(-7)) = 2.7 \\ 0.5\times 3 + 0 + 2\times 0.8\times(9-(-8)) = 28.7 \end{cases} = (2.7, 28.7) \\ x_3 = x_3 + v_3 = (-7,-8) + (2.7, 28.7) = (-4.3, 20.7) \rightarrow (-4.3, 10) \end{cases}$$

对于越界的位置，需要进行合法性调整

粒子群算法实例

- 步骤⑤-⑥：

⑤ 评估粒子的适应度函数值

⑥ 更新粒子的历史最优位置和全局的最优位置

$$f_1' = 1.7^2 - (-1.2)^2 = 1.45 < 39 = f_1$$

$$\begin{cases} f_1 = f_1' = 1.45 \\ pBest_1 = x_1 = (1.7,\ -1.2) \end{cases}$$

$$f_2' = -6.5^2 - 8^2 = -21.75 > -56$$

$$\begin{cases} f_2 = -56 \\ pBest_2 = (-5,\ 9) \end{cases}$$

$$f_3' = (-4.3)^2 - (10)^2 = -81.51 < 15 = f_3$$

$$\begin{cases} f_3 = f_3' = -81.51 \\ pBest_3 = x_3 = (-4.3,10) \end{cases}$$

$$gBest = pBest_3 = (-4.3,10)$$

如果满足结束条件，则输出全局最优结果

并结束程序，否则，转向步骤④继续执行

粒子群优化算法

For each *particle*
    Initialize *particle*
Do
    For each *particle*
        Calculate fitness value
        If fitness value > best fitness value (*pBest*) in history
            set fitness value as new *pBest*
    Choose *particle* with best fitness value of all particles as *gBest*
    For each *particle*
        Calculate *particle* velocity
        Update *particle* position
While maximum iterations or minimum error criteria is not attained

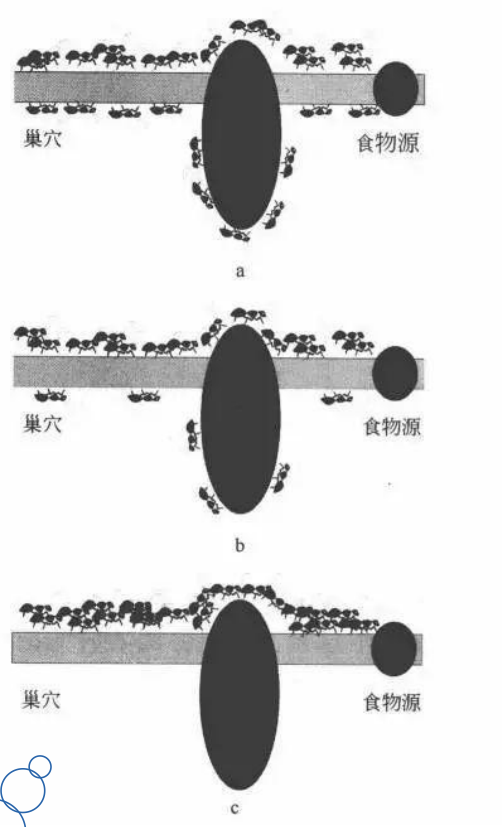
蚁群算法（Ant Colony Optimization， ACO）

- 由意大利科学家Marco Dorigo等人于1991年提出，其灵感来源于蚁群在觅食过程中，总能找到蚁穴到食物之间最短路径的现象

- 在生活中也经常能遇到类似的情况
  - 一群蚂蚁看似在“乱走”
  - 一旦遇到食物，很快就能看到蚂蚁排起队来去寻找食物
  - 蚂蚁是怎么找到这条路的呢?

蚂蚁觅食

- 学者们很早就对蚁群觅食行为进行了生物学研究，发现许多有意思的现象：
  - 蚂蚁“智能”程度非常低，单个蚂蚁的觅食路线具有很大随机性
  - 蚂蚁在外出觅食路上，会沿途释放一种化学物质“信息素”
  - 蚂蚁在遇到多条道路选择时，会根据路径上信息素的浓度来选择道路，信息素浓度高的道路具有更多选择概率
  - 信息素会随着经过路线蚂蚁数量提高而增加，也会随着时间推移逐渐消散



就是这样的简单原则，
使得大量蚂蚁整体表现
出优秀的“寻路”能力

建立蚁群算法

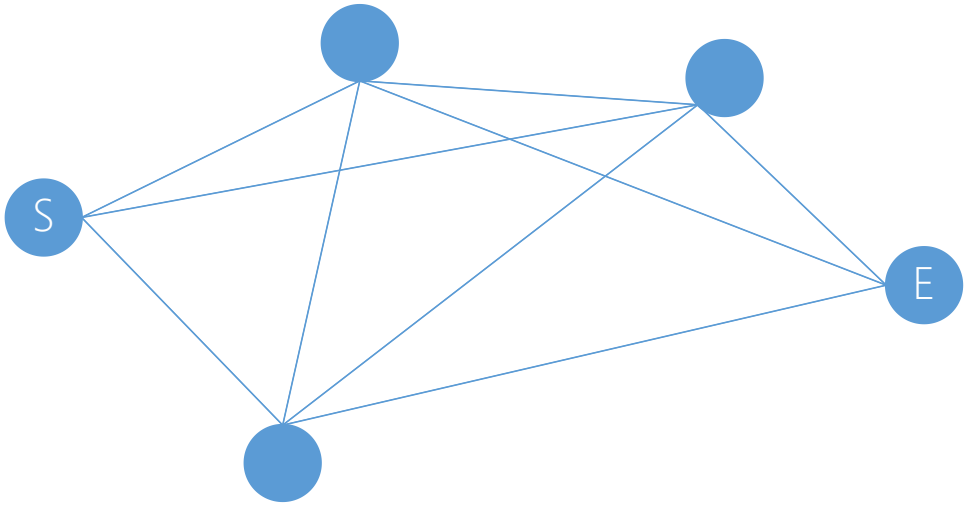
- 要模拟蚁群觅食，需要做四个方面的抽象：
  ① 如何模拟蚂蚁，以及其行为
  ② 如何模拟蚂蚁觅食的环境
  ③ 如何模拟蚂蚁选择路线的行为
  ④ 如何模拟蚂蚁释放信息素、信息素消散的现象

① 模拟蚂蚁

- 蚁群算法中，蚂蚁是最基本单位，不妨称为“人工蚂蚁”，人工蚂蚁应该具有下面的特点：
  - 每只蚂蚁都有相同的目标，以相同速度运动
  - 蚂蚁在行走过程中，在达到目的地前，不走回头路，不转圈
  - 每只蚂蚁都根据相同的原则释放信息素、选择路径
  - 每只蚂蚁都记得自己经历路径的长度和过程
  - 种群中蚂蚁的数量不会发生变化，用m 表示

② 模拟环境：地图

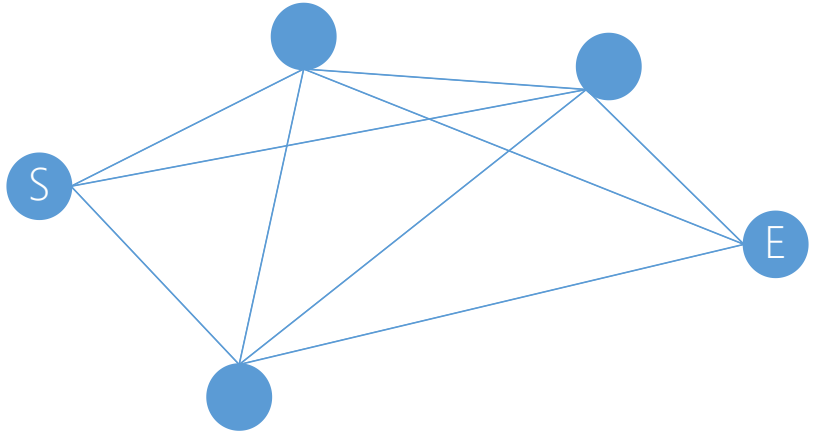
- 将蚂蚁觅食的环境抽象成“具有N个节点的全连通图”
  - 图上一共有N个节点
  - 每个节点都与其他所有节点直接相连，任意两点x，y之间的距离记为d_{xy}均为已知
    （这一点与真实蚂蚁不同）
  - 具有明确的起点和终点

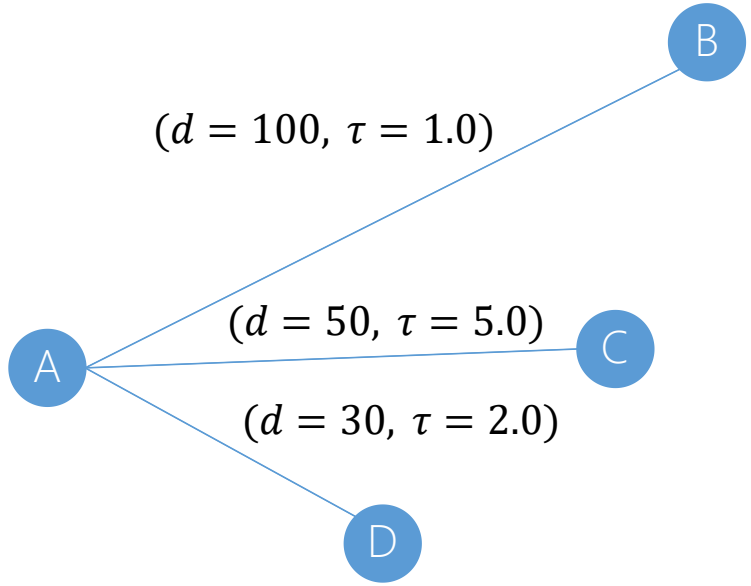


- 如图，图中有5个节点
- S为起点，E为终点
- 蚂蚁总是从起点S出发，到达终点E即结束

③ 模拟蚂蚁选择路线

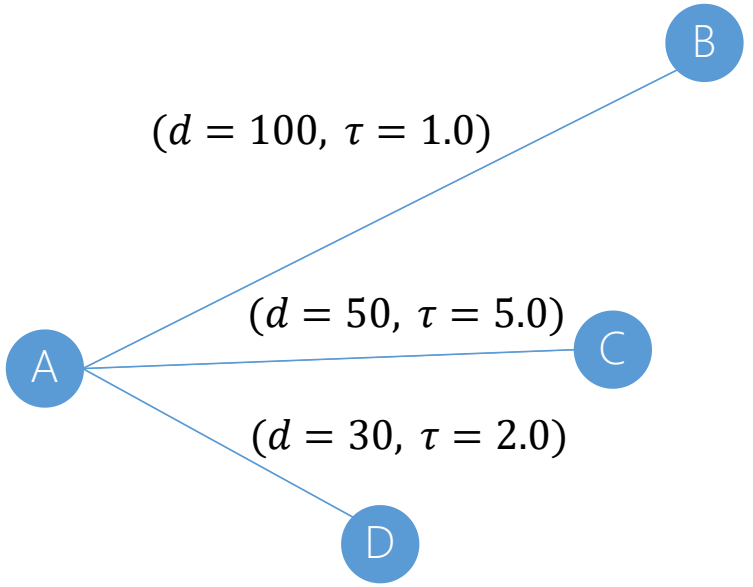
- 有了人工蚂蚁和地图，那么蚂蚁如何选择路线？
  - 假设蚂蚁现在处于A点，下一步有B、C、D三种路线选择
  - 已知路线距离分别为$d_{AB}=100$、$d_{AC}=50$、$d_{AD}=30$
  - 已知路线上的“信息素”$\tau_{AB}=1.0$、$\tau_{AC}=5.0$、$\tau_{AD}=2.0$
  - 那么蚂蚁应该如何选择路线？





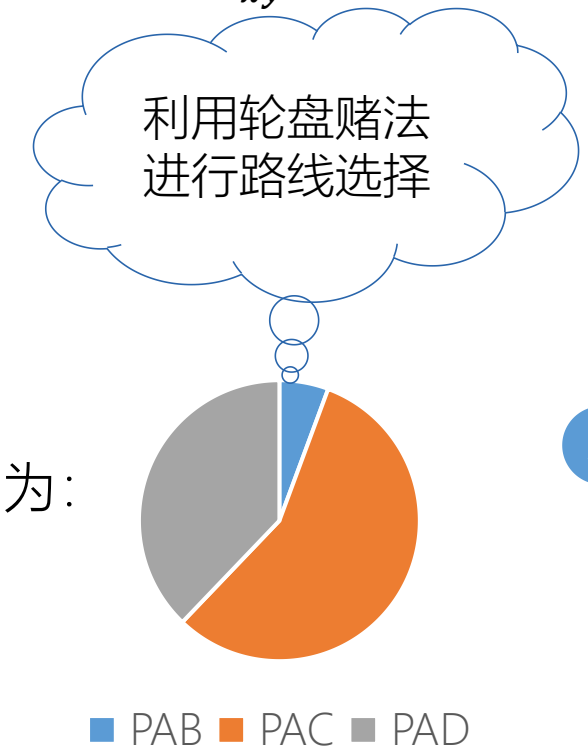
③ 模拟蚂蚁选择路线

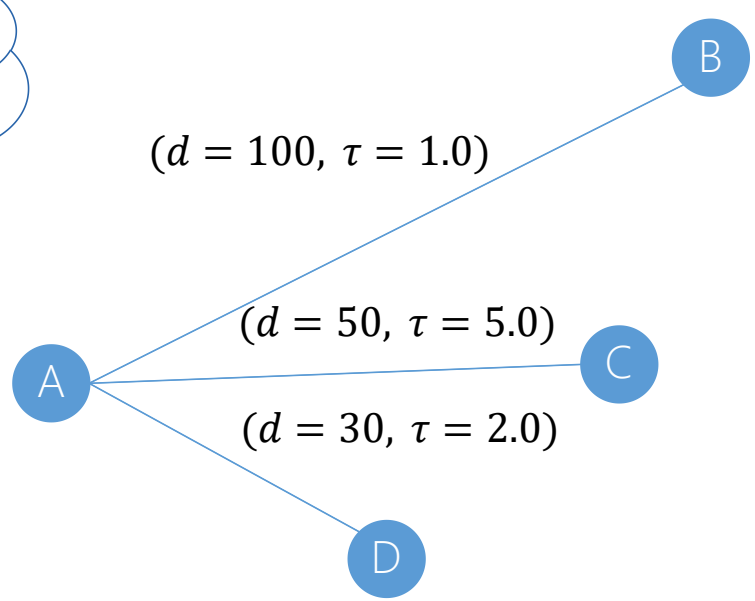
- 第一种情况，不考虑信息素的影响，此时：
  - 人工蚂蚁仅仅知道从A点出发到其余各点的距离
  - 蚂蚁应该从经济性角度出发，选择距离最短的路线行动
- 第二种情况，不考虑各点间的距离，此时：
  - 蚂蚁应该选择“信息素”值最大的路线行动
- 由于蚂蚁智力有限，上面的准则应该概率化，有：
  - 蚂蚁选择路线的概率，与该路线的长度成反比，路线越短，选择概率越高
  - 蚂蚁选择路线的概率，与该路线上信息素浓度成正比，信息素越高，选择概率越高



③ 模拟蚂蚁选择路线

- 因此需要将两方面因素叠加，可以得到概率关系式：$P_{xy} \propto \frac{\tau_{xy}}{d_{xy}}$
- 对于有多种选择的情况，就可以把每个路线上的比值$\frac{\tau_{xy}}{d_{xy}}$求出来，做概率化
- 三条路线对应的分数分别为：
  - $\frac{\tau_{AB}}{d_{AB}} = 0.01$、$\frac{\tau_{AC}}{d_{AC}} = 0.1$、$\frac{\tau_{AD}}{d_{AD}} = 0.067$
- 则三条路线被选择的概率（转移概率）分别为：
  - $P_{AB} = \frac{0.01}{0.01+0.1+0.067} = 0.056$
  - $P_{AC} = 0.57$
  - $P_{AD} = 0.374$





③ 模拟蚂蚁选择路线

- 为了进一步提高灵活性，将$\eta_{xy}=\frac{1}{d_{xy}}$称为"能见度因子"，然后为$\tau_{xy}$和$\eta_{xy}$添加指数权重$\alpha$和$\beta$，得到最终的概率计算式：

$$P_{xy}=\begin{cases}\dfrac{[\tau_{xy}]^{\alpha}[\eta_{xy}]^{\beta}}{\sum_i[\tau_{xi}]^{\alpha}[\eta_{xi}]^{\beta}}, i\in allowed(y)\\ 0,\ otherwise\end{cases}$$

已经走过的节点被存在禁忌表中，之后是不可以被访问的

- 分母是从x出发的所有合理路径上，能见度和信息素得分累积
- 分子是从x到y路径上的能见度和信息素得分
- 仅对合理路径进行计算，不合理路径包括原路返回、转圈、再次经过同一条路线等

- 信息素多的地方显然经过这里的蚂蚁多，因而会有更多的蚂蚁聚集过来

- 正反馈现象：某一路径上走过的蚂蚁越多，则后来者选择该路径的概率就越大

④ 模拟信息素释放和消散

- 在实际中，每个蚂蚁独立行动，边运动边释放信息素，同时信息素随着时间推移而消散，整个过程与时间相关。

- 可以在系统中添加真实时间如秒、分钟等，来精确模拟每个蚂蚁在一定时间段中的行为，但由于研究的是蚁群整体，因此这种精细模拟并非必要。

- 一种常见的策略，是将时间“**片段化**”，以“cycle”为单位来模拟时间
  - 一个cycle表示蚁群中所有蚂蚁均从出发点成功达到目标点
  - 信息素在一个cycle结束之后统一更新，不考虑cycle期间信息素的消散和累积

④ 模拟信息素释放和消散

- 这种假设称为“蚁圈模型”
- 蚁圈模型中，以cycle为时间单位，每个cycle结束后，时间t增加1
- 为了模拟信息素的消散，简单认为信息素随时间以一定比例逐渐消散，这个比例设计成“信息素保持系数”ρ，有$0 < \rho < 1$
- 对于路径xy而言，如果没有蚂蚁经过则：
  - $\tau_{xy}(t+1) = \rho \cdot \tau_{xy}(t)$
- 由于$\rho < 1$，若xy路径始终没有蚂蚁经过，则该路径上的信息素会逐渐减少，直至消散为零

④ 模拟信息素释放和消散

- 如果在t对应的cycle中，有若干蚂蚁经过了xy路径，则应该对xy上的信息素有所增加，将这个信息素增量记为$\Delta\tau_{xy}(t)$
- 假设有k只蚂蚁，在t对应的cycle经过了xy路径，则xy路径下一个cycle对应的信息量应该为：
  - $\tau_{xy}(t+1) = \rho \cdot \tau_{xy}(t) + [\Delta\tau_{xy}^{1}(t) + \Delta\tau_{xy}^{2}(t) + \ldots + \Delta\tau_{xy}^{k}(t)]$
- 那么问题就再次转化为：每只蚂蚁对路径xy的信息素增量$\Delta\tau_{xy}^{k}(t)$是多少？

④ 模拟信息素释放和消散

- 对于一只蚂蚁，在一个cycle结束之后，其经过的路径是可以记录的
- 找到比较好的路线的蚂蚁应该很“优秀”，因此它留下的信息素更有价值
- 即如果蚂蚁k找到的路线越短，那么沿途留下的信息素应该越多
- 若蚂蚁k经过了xy，那它留下的信息素就是：$\Delta\tau_{xy}^{k}(t)=\frac{Q}{L_k}$
- 其中Q为常数，L_k就是蚂蚁k在当前cycle中找到的路径的总长度

蚁群算法实施步骤

将m 只蚂蚁随机放置，设定信息素初始值

将当前位置添加到禁忌表

根据转移概率选择下一地点

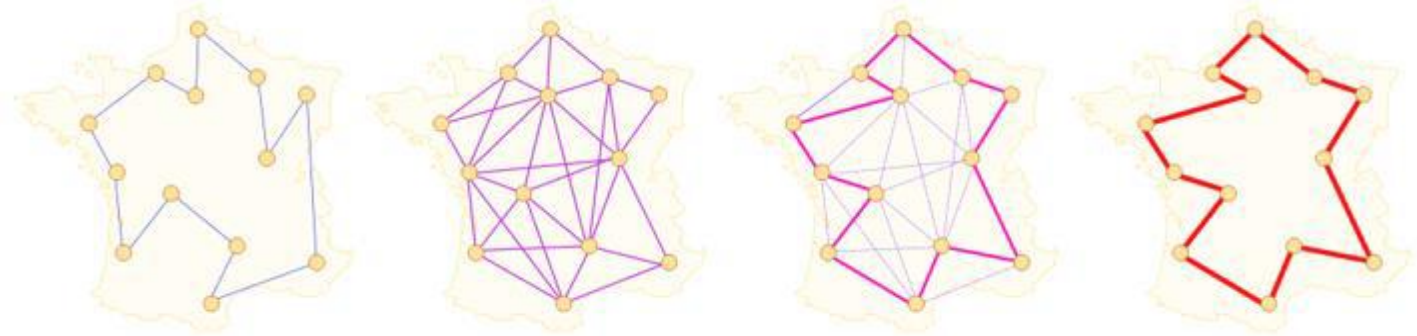
每只蚂蚁周游一周之后，计算每条边上信息素增量

判断周游次数是否满足停机准则

例：利用蚁群算法解决旅行商问题

旅行商问题

- 旅行商问题(Traveling Salesman Problem, TSP)是数学领域中著名问题之一
  - 假设有一个旅行商人要拜访n个城市，路径的限制是每个城市只能拜访一次，而且最后要回到原来出发的城市
  - 路径的选择目标：路径的路程最小
- 数据集：51个城市的坐标（经纬度表示）
- 约束：访问所有节点，每个节点有且仅有一次被访问
- 目标：访问的路径最小



蚁群算法解决旅行商问题的主要思想及步骤

- 主要思想
  - 每只蚂蚁都知道任意两个城市的距离
  - 每只蚂蚁会随机选择一个起点出发
  - 每只蚂蚁会有较大概率选择信息素浓度高且路程短的路径
  - 蚂蚁经过的路径都会留下信息素
  - 当所有蚂蚁周游一圈后，会聚在一起相互交流信息

主要思想：步骤④前，每只蚂蚁选择路径依靠自己（个体智慧），步骤④是将每只蚂蚁的个体信息聚合为群体信息（群体智慧），并指导所有蚂蚁下一步的行动

- 主要步骤
  - ① 导入51个城市的经纬度数据并计算城市间相互距离（距离矩阵）
  - ② 初始化参数（蚂蚁种群规模，最大迭代次数，初始状态转移矩阵（由信息素与距离决定）等）
  - ③ 根据状态转移矩阵确定每只蚂蚁的初始路径（每只蚂蚁随机选择初始起点城市）
  - ④ 待所有蚂蚁都完成初始路径后，更新状态转移矩阵
  - ⑤ 重复步骤③，直至达到停止准则，并输出最短路径及最短路径的总路程

搜索算法小结

✓ **Local search:**

Hill-Climbing operate on complete-state formulations; Local Beam keeps track of k states; Tabu Search uses a neighborhood search with some constraints.

局部搜索：

爬山法在完整状态形式化上进行操作；局部束搜索法保持k个状态的轨迹；禁忌搜索采用一种带约束的邻域搜索。

✓ **Optimization and Evolutionary Algorithms:**

Simulated Annealing approximate global optimization in a large search space; Genetic Algorithm mimics the evolutional process of natural selection.

优化与进化算法：

模拟退火在大搜索空间逼近全局最优解；遗传算法模仿自然选择的进化过程。

✓ **Swarm Intelligence:**

Ant Colony Optimization can be reduced to finding good paths through graphs; Particle Swarm Optimization is by iteratively trying to improve a candidate solution.

群体智能：

蚁群优化可以寻找图的最好路径；粒子群优化通过迭代来改善一个候选解。