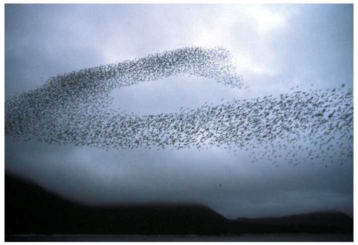
人工智能概论

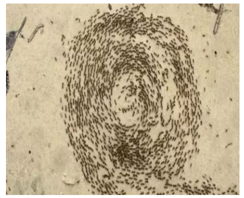
西安电子科技大学 人工智能学院

第九章 粒子群算法

鸟群迁徙/觅食场景

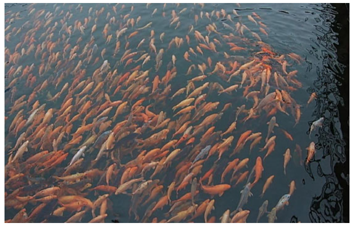


蚁群觅食场景



在一个复杂的生态系统中，蚁群以其卓越的协作能力和信息传递机制，形成了一个高效的觅食场景。

鱼群觅食场景



- 群智能概念

  - “群智能”是指由简单个体组成的群落与环境以及个体之间的互动行为。
  - 群智能中的群，可被定义为“一组相互之间可以进行直接或间接通信的主体”。群的个体组织包括在结构上很简单的鸟群、蚁群、鱼群、蜂群等，而它们的集体行为却可能变得相当复杂。

- 群智能算法基本思想

  - 群智能算法的基本思想是模拟自然界生物的群体行为来构造随机优化算法

自然界	群智能算法
个体	搜索空间中的点
个体的进化或觅食过程	搜索和优化过程
个体对环境的适应能力	求解问题的目标函数
个体的优胜劣汰过程或觅食过程	可行解的迭代过程

  - 群智能算法是一种具有“生成+检验”特征的迭代搜索算法

- 常见的群智能算法

  - 目前群智能算法已包括三十余种，常见的群智能算法有：
    - 粒子群优化算法
    - 蚁群优化算法
    - 菌群算法
    - 人工鱼群算法
    - 狼群算法
    - 布谷鸟搜索算法
    - ······

粒子群算法原理

粒子群算法背景

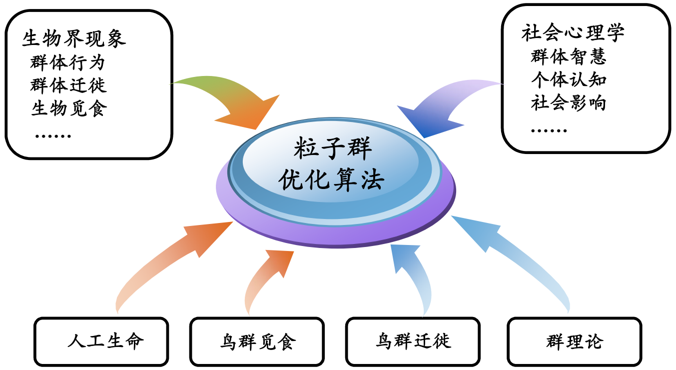
- 粒子群算法的提出
  - 1995年，受到鸟群活动的规律性启发，Russell Eberhart和James Kennedy建立了一个简化模型，并最终形成了粒子群优化算法 (Particle Swarm Optimization, PSO)，也可称为粒子群算法。

鸟类群体捕食

Russell Eberhart
电子工程学博士

James Kennedy
社会心理学博士

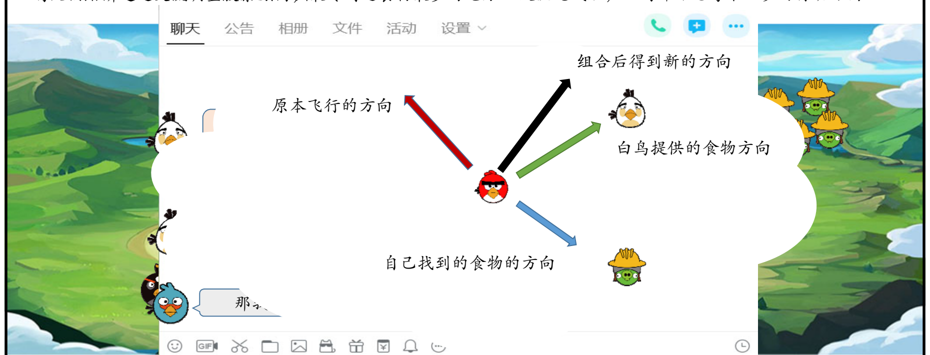
粒子群算法背景



粒子群算法原理

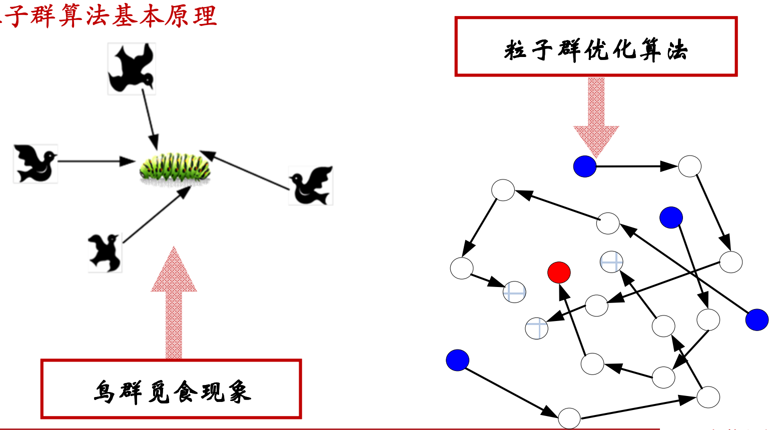
- 粒子群算法基本原理

物最多的地方　交流完成后，红鸟开始思考下一步的行动方向



粒子群算法原理

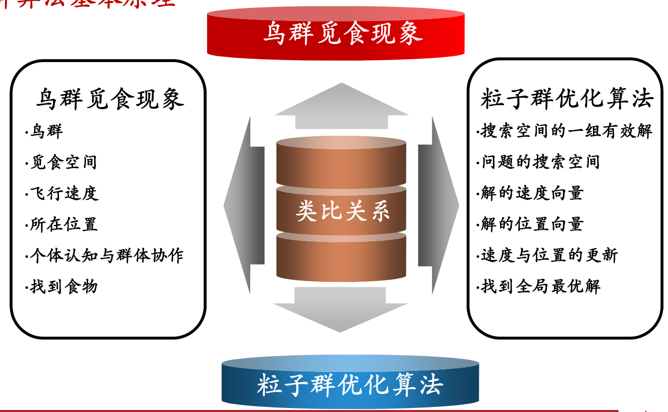
- 粒子群算法基本原理



西安电子科技大学 XIDIAN UNIVERSITY

粒子群算法原理

- 粒子群算法基本原理



西安电子科技大学 XIDIAN UNIVERSITY

粒子群算法原理

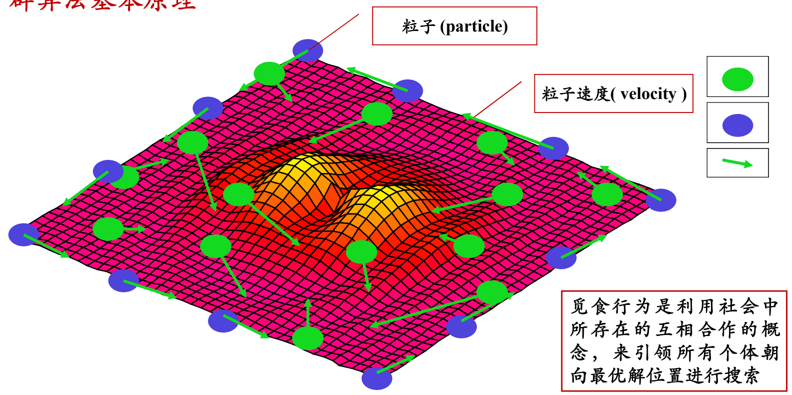
- 粒子群算法基本原理

  - 在PSO中，每个解看作一只鸟，称为粒子(particle)，所有的粒子都有一个适应值，每个粒子都有一个速度决定它们的飞翔方向和距离，粒子们追随当前最优粒子在解空间中搜索。
  - 当其它鸟发现了更佳的觅食地点，鸟群间会有某种类似广播的沟通行为，渐渐的将其它鸟群引领至较佳的地点。这样的觅食行为是利用社会中所存在的互相影响的概念，来引领所有个体朝向最佳解位置。

西安电子科技大学 XIDIAN UNIVERSITY

粒子群算法原理

- 粒子群算法基本原理



- 粒子群算法基本原理

假设在D维搜索空间中，有m个粒子。

- (1) 其中第i个粒子的位置向量表示为：

$$x_i=(x_{i1},x_{i2},\cdots,x_{iD})$$

- (2) 其飞行速度向量表示为：

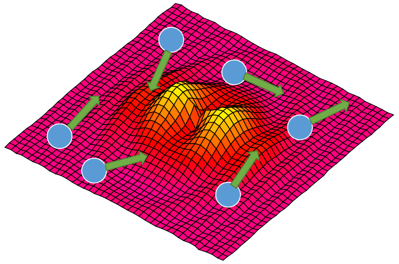
$$v_i=(v_{i1},v_{i2},\cdots,v_{iD})$$

- (3) 第i个粒子搜索到的最优位置为：

$$pBest_i=(pBest_{i1},pBest_{i2},\cdots,pBest_{iD})$$

- (4) 整个粒子群搜索到的最优位置为：

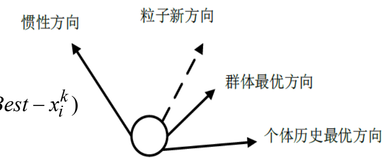
$$gBest=(gBest_1,gBest_2,\cdots,gBest_D)$$



- 粒子速度和位置的更新

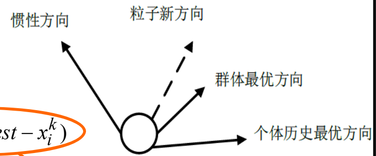
$$v_i^{k+1}=wv_i^k+c_1rand_1(pBest_i-x_i^k)+c_2rand_2(gBest-x_i^k)$$

$$x_i^{k+1}=x_i^k+v_i^{k+1}\qquad i=1,2,\cdots,m;$$



- w为惯性权重
- c_1和c_2为两个正常数，称为加速因子；$rand_1$和$rand_2$为均匀分布于[0,1]的随机数

- 粒子速度和位置的更新



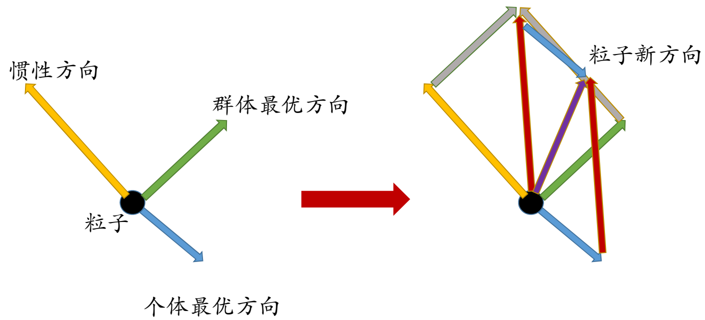
$$v_i^{k+1}=wv_i^k+c_1rand_1(pBest_i-x_i^k)+c_2rand_2(gBest-x_i^k)$$

$$x_i^{k+1}=x_i^k+v_i^{k+1}\qquad i=1,2,\cdots,m;$$

“惯性部分”：对自身运动状态的信任

“认知部分”：对粒子本身的思考，即来源于自己经验的部分

“社会部分”：粒子间的信息共享，来源于群体中的其它优秀粒子的经验



粒子群算法原理

➢ 超界处理：

- **随机初始化法**：从搜索空间或速度范围中产生一个随机数代替超出范围的数值。以x_s和v_s表示已经飞出边界的第s维的位置或速度，x_s'和v_s'表示其新的数值，则：

$$\begin{aligned} x_s' &= rand(x_{\min}, x_{\max}) \\ v_s' &= rand(v_{\min}, v_{\max}) \end{aligned}$$

- **反弹法**：假设边界是一堵有弹力的墙，粒子某维如欲飞出边界,则被弹回。若飞出边界后可继续前行l距离，则被边界弹回的距离亦为l。x_s'和v_s'表达如下：

西安电子科技大学 XIDIAN UNIVERSITY

粒子群算法原理

➢ 超界处理：

- 反弹法：

$$\begin{aligned} x_s' &= 2*x_{max} - x_s \quad if\ x_s > x_{max} \\ x_s' &= 2*x_{min} - x_s \quad if\ x_s < x_{min} \\ v_s' &= 2*v_{max} - v_s \quad if\ v_s > v_{max} \\ v_s' &= 2*v_{min} - v_s \quad if\ v_s < v_{min} \end{aligned}$$

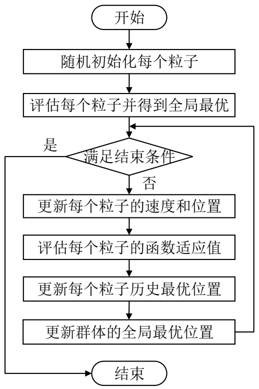
$$\begin{aligned} x_s' &= x_{max} \quad if\ x_s > x_{max} \\ x_s' &= x_{min} \quad if\ x_s < x_{min} \\ v_s' &= v_{max} \quad if\ v_s > v_{max} \\ v_s' &= v_{min} \quad if\ v_s < v_{min} \end{aligned}$$

- **粘滞法**：直接将飞出边界的数值拉回到边界之上使之不再继续飞行，则：

西安电子科技大学 XIDIAN UNIVERSITY

粒子群算法流程

- PSO算法流程图和伪代码




```
//功能：粒子群优化算法伪代码
//说明：本例以求问题最小值为目标
//参数：N为群体规模

procedure PSO
   for each particle i
      Initialize velocity Vi and position Xi for particle i
      Evaluate particle i and set pBesti = Xi
   end for
   gBest = min {pBesti}
   while not stop
      for i=1 to N
         Update the velocity  and position of particle i
         Evaluate particle i
         if fit (Xi) < fit (pBesti)
             pBesti = Xi;
         if fit(pBesti) < fit (gBest)
             gBest = pBesti;
       end for
   end while
   print gBest
end procedure
```


西安电子科技大学 XIDIAN UNIVERSITY

粒子群算法原理

- 思考：粒子在进行速度更新时，受到哪些因素的影响？

西安电子科技大学 XIDIAN UNIVERSITY

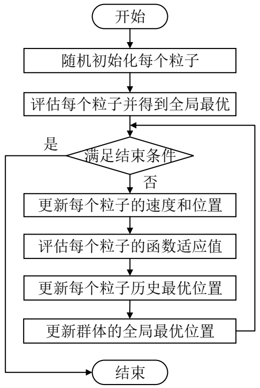
粒子群算法求解实例

- 粒子群算法求解实例--连续优化问题

已知函数 $y=f(x_1,x_2)=x_1^2+x_2^2$

其中 $-10\le x_1,x_2\le 10$

用粒子群优化算法求解y的最小值。



西安电子科技大学 XIDIAN UNIVERSITY

粒子群算法求解实例

- 粒子群算法求解实例--连续优化问题

$y=f(x_1,x_2)=x_1^2+x_2^2$

$-10\le x_1,x_2\le 10$

➢ 粒子群算法的参数设置

参数名称	具体数值
粒子群大小 m	3
惯性权重 w	0.5
加速系数 c_1 c_2	2
r_1 r_2	[0,1]随机数
自变量维数	2
各维自变量最小值	-10
各维自变量最大值	10

西安电子科技大学 XIDIAN UNIVERSITY

粒子群算法求解实例

- 粒子群算法求解实例--连续优化问题

$y=f(x_1,x_2)=x_1^2+x_2^2$

$-10\le x_1,x_2\le 10$

➢ Step1：初始化

▪ 粒子群大小m=3；在搜索空间中随机初始化每个解的速度和位置

$$p_1=\begin{cases}v_1=(3,2)\\x_1=(8,-5)\end{cases}\quad p_2=\begin{cases}v_2=(-3,-2)\\x_2=(-5,9)\end{cases}\quad p_3=\begin{cases}v_3=(5,3)\\x_3=(-7,-8)\end{cases}$$

西安电子科技大学 XIDIAN UNIVERSITY

粒子群算法求解实例

- 粒子群算法求解实例--连续优化问题

$y=f(x_1,x_2)=x_1^2+x_2^2$

$-10\le x_1,x_2\le 10$

➢ Step2：粒子评估

▪ 计算适应函数值，并且得到粒子的历史最优位置和群体的全局最优位置。

$$p_1=\begin{cases}v_1=(3,2)\\x_1=(8,-5)\end{cases}\quad \begin{cases}f_1=8^2+(-5)^2=64+25=89\\ pBest_1=x_1=(8,-5)\end{cases}$$

直接用目标函数作为适应度值！

$$p_2=\begin{cases}v_2=(-3,-2)\\x_2=(-5,9)\end{cases}\quad \begin{cases}f_2=(-5)^2+9^2=25+81=106\\ pBest_2=x_2=(-5,9)\end{cases}$$

$$gBest=pBest_1=(8,-5)$$

$$p_3=\begin{cases}v_3=(5,3)\\x_3=(-7,-8)\end{cases}\quad \begin{cases}f_3=(-7)^2+(-8)^2=49+64=113\\ pBest_3=x_3=(-7,-8)\end{cases}$$

西安电子科技大学 XIDIAN UNIVERSITY

粒子群算法求解实例

- 粒子群算法求解实例--连续优化问题
  - Step3：更新粒子速度和位置

$$v_i^{k+1}=wv_i^k+c_1rand_1(pBest_i-x_i^k)+c_2rand_2(gBest-x_i^k)$$
$$x_i^{k+1}=x_i^k+v_i^{k+1}\qquad i=1,2,\cdots,m;$$

- 根据自身的历史最优位置和全局的最优位置，更新每个粒子的速度和位置。

个体最优初始条件下默认是当前位置

$$p_1=\begin{cases}v_1=(3,2)\\x_1=(8,-5)\end{cases}\qquad p_1=\begin{cases}v_1=\omega\times v_1+c_1\times r_1\times(pBest_1-x_1)+c_2\times r_2\times(gBest-x_1)\\ \Rightarrow v_1=\begin{cases}0.5\times3+0+0=1.5\\0.5\times2+0+0=1\end{cases}=(1.5,1)\\x_1=x_1+v_1=(8,-5)+(1.5,1)=(9.5,-4)\end{cases}$$

ω=0.5, c1=c2=2, r1,r2是[0,1]的随机数

西安电子科技大学 XIDIAN UNIVERSITY

粒子群算法求解实例

- 粒子群算法求解实例--连续优化问题
  - Step3：更新粒子速度和位置

$$v_i^{k+1}=wv_i^k+c_1rand_1(pBest_i-x_i^k)+c_2rand_2(gBest-x_i^k)$$
$$x_i^{k+1}=x_i^k+v_i^{k+1}\qquad i=1,2,\cdots,m;$$

- 根据自身的历史最优位置和全局的最优位置，更新每个粒子的速度和位置。

$$p_2=\begin{cases}v_2=(-3,-2)\\x_2=(-5,9)\end{cases}\qquad p_2=\begin{cases}v_2=\omega\times v_2+c_1\times r_1\times(pBest_2-x_2)+c_2\times r_2\times(gBest-x_2)\\ \Rightarrow v_2=\begin{cases}0.5\times(-3)+0+2\times0.3\times(8-(-5))=6.3\\0.5\times(-2)+0+2\times0.1\times((-5)-9)=-3.8\end{cases}=(6.3,-3.8)\\x_1=x_1+v_1=(-5,9)+(6.3,-3.8)=(1.3,5.2)\end{cases}$$

ω=0.5, c1=c2=2, r1,r2是[0,1]的随机数

西安电子科技大学 XIDIAN UNIVERSITY

粒子群算法求解实例

- 粒子群算法求解实例--连续优化问题
  - Step3：更新粒子速度和位置

$$v_i^{k+1}=wv_i^k+c_1rand_1(pBest_i-x_i^k)+c_2rand_2(gBest-x_i^k)$$
$$x_i^{k+1}=x_i^k+v_i^{k+1}\qquad i=1,2,\cdots,m;$$

- 根据自身的历史最优位置和全局的最优位置，更新每个粒子的速度和位置。

$$p_3=\begin{cases}v_3=(5,3)\\x_3=(-7,-8)\end{cases}\qquad p_3=\begin{cases}v_3=\omega\times v_3+c_1\times r_1\times(pBest_3-x_3)+c_2\times r_2\times(gBest-x_3)\\ \Rightarrow v_3=\begin{cases}0.5\times5+0+2\times0.05\times(8-(-7))=4\\0.5\times3+0+2\times0.8\times((-5)-(-8))=6.3\end{cases}=(4,6.3)\\x_1=x_1+v_1=(-7,-8)+(4,6.3)=(-3,-1.7)\end{cases}$$

ω=0.5, c1=c2=2, r1,r2是[0,1]的随机数

西安电子科技大学 XIDIAN UNIVERSITY

粒子群算法求解实例

- 粒子群算法求解实例--连续优化问题
  - Step4：粒子评估

$$y=f(x_1,x_2)=x_1^2+x_2^2$$
$$-10\le x_1,x_2\le10$$

- 计算适应函数值，并且得到粒子的历史最优位置和群体的全局最优位置。

$$p_1=\begin{cases}v_1=(1.5,1)\\x_1=(9.5,-4)\end{cases}\qquad f_1^*=9.5^2+(-4)^2=90.25+16=106.25>f_1=89\qquad \boldsymbol{pBest}_1=\boldsymbol{x}_1=(8,-5)$$

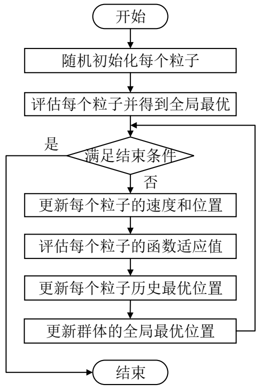
$$p_2=\begin{cases}v_2=(6.3,-3.8)\\x_2=(1.3,5.2)\end{cases}\qquad f_2^*=1.3^2+5.2^2=1.69+27.04=31.94<106=f_2\qquad \boldsymbol{pBest}_2=(1.5,3.2)$$

$$p_3=\begin{cases}v_3=(4,6.3)\\x_3=(-3,-1.7)\end{cases}\qquad f_3^*=(-3)^2+(-1.7)^2=9+2.89=11.89<113=f_3\qquad \boldsymbol{pBest}_3=(-3,-1.7)$$

$$\boldsymbol{gBest}=\boldsymbol{pBest}_3=(-3,-1.7)$$

粒子群算法求解实例

- 粒子群算法求解实例--连续优化问题
  - Step5：终止条件判断
    - 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Step2继续执行。



粒子群算法参数设置

- 粒子群算法参数设置
  - （1）惯性权重w

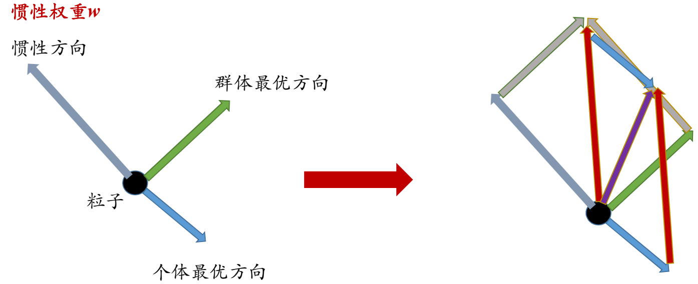
$$v_i^{k+1} = wv_i^k + c_1 rand_1(pBest_i - x_i^k) + c_2 rand_2(gBest - x_i^k)$$
$$x_i^{k+1} = x_i^k + v_i^{k+1} \qquad i = 1,2,\cdots,m;$$

- 使粒子保持运动惯性，使其有搜索扩展空间的趋势，有能力探索新的区域。
- 也表示粒子对当前自身运动状态的信任，依据自身的速度进行惯性运动。
- 较大的w有利于跳出局部极值，而较小的w有利于算法收敛。

粒子群算法参数设置

- 粒子群算法参数设置
  - 惯性权重w

速度更新示意图

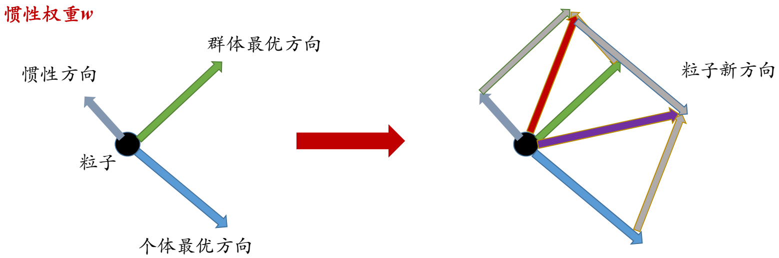


惯性速度大的粒子更趋向于探索未知的空间，保证算法的探索能力

粒子群算法参数设置

- 粒子群算法参数设置
  - 惯性权重w

速度更新示意图



惯性速度小的粒子更趋向于跟随种群最优方向，保证算法的收敛能力

西安电子科技大学 XIDIAN UNIVERSITY

粒子群算法参数设置

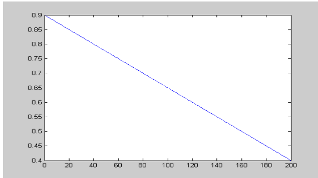
- 粒子群算法参数设置

$$v_i^{k+1} = wv_i^k + c_1 rand_1(pBest_i - x_i^k) + c_2 rand_2(gBest - x_i^k)$$
$$x_i^{k+1} = x_i^k + v_i^{k+1} \qquad i = 1, 2, \cdots, m;$$

➢ 惯性权重w改进

■ 自适应调整策略：随着迭代的进行，线性地减小 w 的值

$$w = w_{\max} - \frac{w_{\max} - w_{\min}}{iter_{\max}} \times iter$$



$w_{\max}$、$w_{\min}$分别是w的最大值和最小值；

iter、$iter_{\max}$分别是当前迭代次数和最大迭代次数；

西安电子科技大学 XIDIAN UNIVERSITY

粒子群算法参数设置

- 粒子群算法参数设置

➢ 加速因子 c_1 c_2

$$v_i^{k+1} = wv_i^k + c_1 rand_1(pBest_i - x_i^k) + c_2 rand_2(gBest - x_i^k)$$
$$x_i^{k+1} = x_i^k + v_i^{k+1} \qquad i = 1, 2, \cdots, m;$$

- 代表将粒子推向*pBest*和*gBest*位置的统计加速项的权重。
- 表示粒子的动作来源于自己经验的部分和其它粒子经验的部分。
- 低的加速因子值使粒子在目标区域外徘徊，而高的加速因子值导致粒子越过目标区域。

西安电子科技大学 XIDIAN UNIVERSITY

粒子群算法参数设置

- 粒子群算法参数设置

➢ 加速因子c_1 c_2的改进

$$v_i^{k+1} = wv_i^k + c_1 rand_1(pBest_i - x_i^k) + c_2 rand_2(gBest - x_i^k)$$
$$x_i^{k+1} = x_i^k + v_i^{k+1} \qquad i = 1, 2, \cdots, m;$$

- 通常将c_1 c_2统一为一个控制参数，$\varphi = c_1 + c_2$；
- 如果φ很小，粒子群运动轨迹将非常缓慢；如果φ很大，则粒子位置变化非常快；
- 通过仿真可以获得φ的经验值，当φ=4.0 (c_1 =2.0, c_2 =2.0) 时，具有很好的收敛效果；

西安电子科技大学 XIDIAN UNIVERSITY

粒子群算法优缺点

- 粒子群算法优缺点

➢ 优点

1. 简单易实现；
2. 收敛速度快；
3. 粒子有记忆功能；

➢ 缺点

1. 缺乏速度的自适应调节，容易陷入局部最优，可能导致收敛精度低或不收敛；
2. 标准粒子群算法不能有效求解离散及组合优化问题；
3. 参数难以确定，对不同的问题需选择合适的参数来达到最优效果；

西安电子科技大学 XIDIAN UNIVERSITY

粒子群算法

- 粒子群算法与遗传算法的共性
  - 都属于仿生算法；
  - 都属于全局优化方法；
  - 都属于随机搜索算法；
  - 都隐含并行性；
  - 根据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；

西安电子科技大学 XIDIAN UNIVERSITY

粒子群算法

- 粒子群算法与遗传算法的差异

粒子群算法	遗传算法
粒子群算法有记忆，所有粒子都保存较优解的知识；	遗传算法中以前的知识随着种群的改变被改变；
粒子群算法中的粒子是一种单向共享信息机制；	遗传算法中的染色体之间相互共享信息；
粒子群算法中粒子只是通过内部速度进行更新，实现更容易；	遗传算法需要编码和遗传操作；

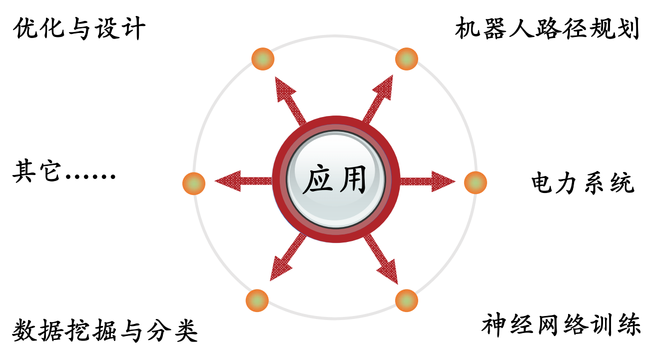
西安电子科技大学 XIDIAN UNIVERSITY

粒子群算法

- 粒子群优化算法的改进研究

西安电子科技大学 XIDIAN UNIVERSITY

粒子群算法应用



- 课后小结
  - PSO研究背景
  - PSO基本原理及模型
- 课后思考
  - PSO算法的优缺点?

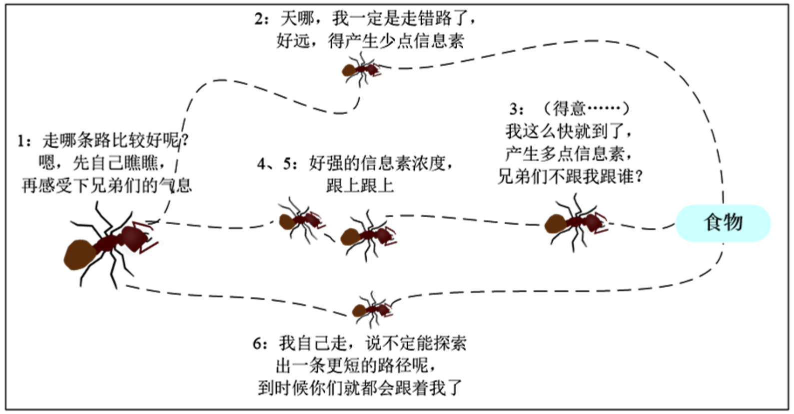
第九章 蚁群算法

9.1 基本原理

9.2 算法流程

9.3 求解实例

基本原理



基本原理

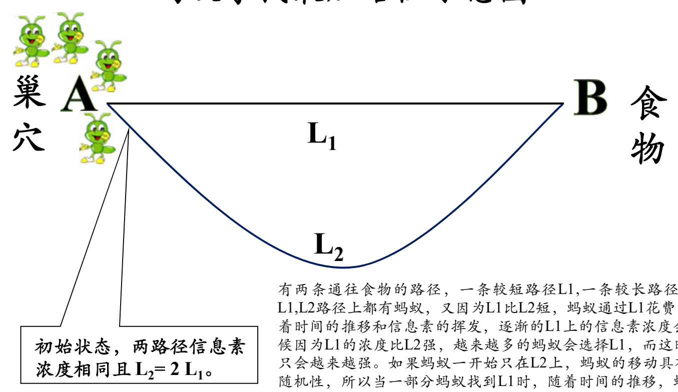
基本原理

- 蚁群靠什么找出最短路径?

1. 信息素（pheromone）：信息素是一种由蚂蚁自身释放的易挥发的物质，能够实现蚁群内的间接通信。蚂蚁在寻找食物时，会在其经过的路上释放信息素，而信息素可以被其它的蚂蚁感知，并且信息素的浓度越高，对应的路径越短。

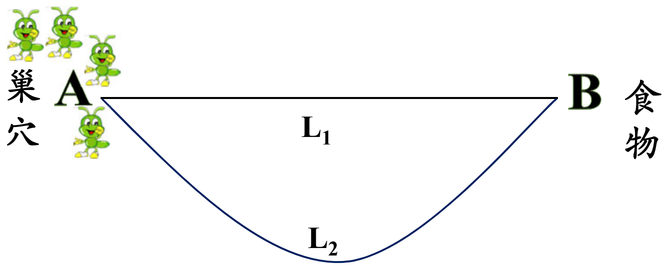
2. 正反馈：蚂蚁会以较大的概率选择信息素浓度较高的路径，并释放一定量的信息素，从而使距离较短路径的信息素浓素被加强，形成一个正反馈。

蚂蚁寻找最短路径示意图

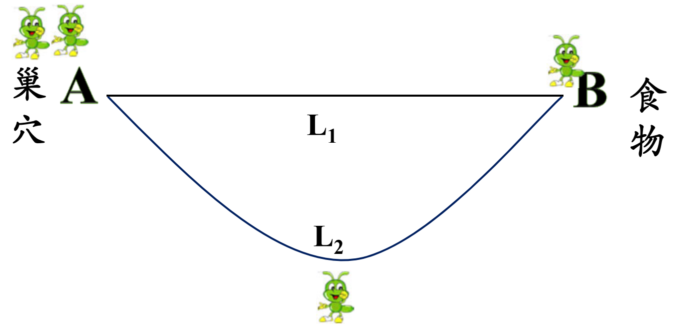


有两条通往食物的路径，一条较短路径L1,一条较长路径L2,虽然刚开始L1,L2路径上都有蚂蚁，又因为L1比L2短，蚂蚁通过L1花费的时间较短，随着时间的推移和信息素的挥发，逐渐的L1上的信息素浓度会强于L2，这时候因为L1的浓度比L2强，越来越多的蚂蚁会选择L1，而这时候L1上的浓度只会越来越强。如果蚂蚁一开始只在L2上，蚂蚁的移动具有一定小概率的随机性，所以当一部分蚂蚁找到L1时，随着时间的推移，蚂蚁会收敛到L1上，从而可以跳出局部最优。

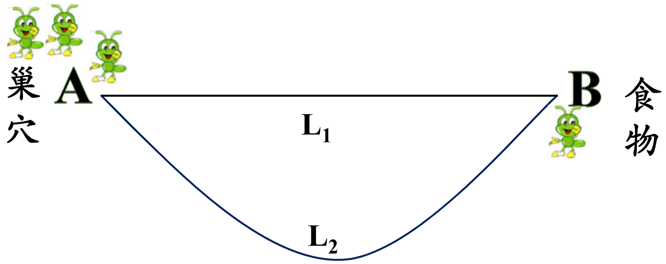
蚂蚁寻找最短路径示意图



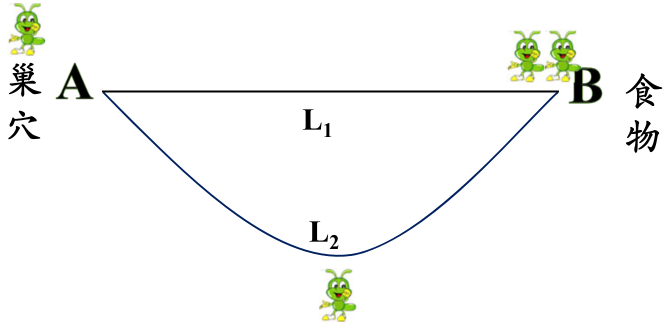
蚂蚁寻找最短路径示意图



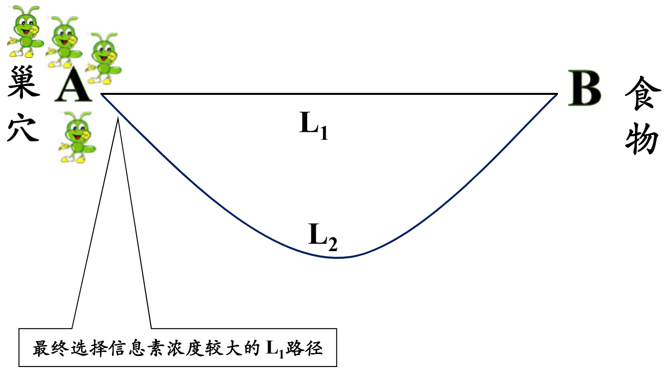
蚂蚁寻找最短路径示意图



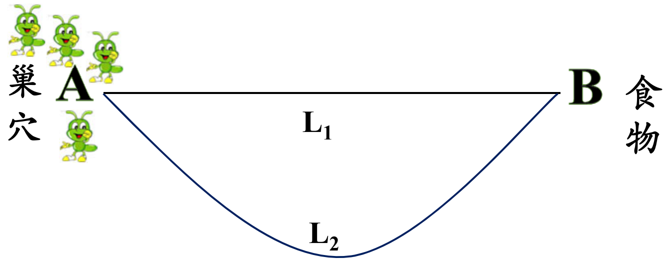
蚂蚁寻找最短路径示意图



蚂蚁寻找最短路径示意图

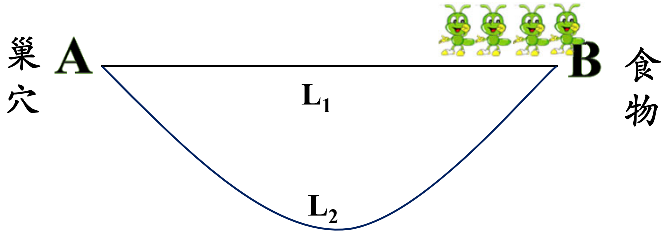


蚂蚁寻找最短路径示意图

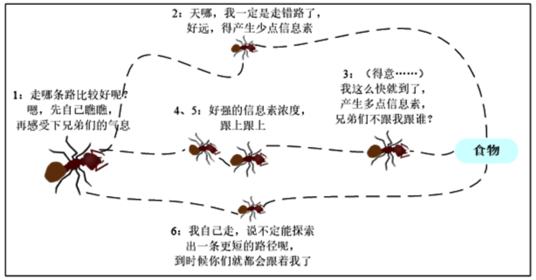


基本原理

蚂蚁寻找最短路径示意图

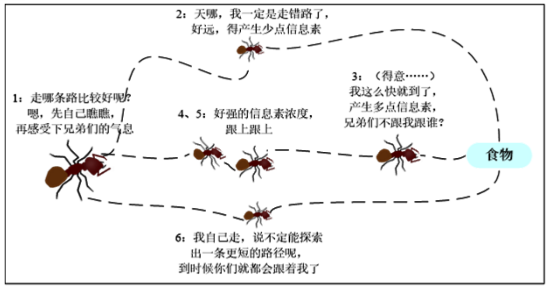


基本原理



蚂蚁在寻找食物的过程中往往是随机选择路径的，但它们能感知当前地面上的信息素浓度，并倾向于往信息素浓度高的方向行进。**由于较短路径上蚂蚁的往返时间比较短，单位时间内经过该路径的蚂蚁多，所以信息素的积累速度比较长的路径快。**

基本原理



因此，当后续蚂蚁在路口时，就能感知先前蚂蚁留下的信息素，并倾向于选择一条较短的路径前行。**这种正反馈机制使得越来越多的蚂蚁在巢穴与食物之间的最短路径上行进。**

基本原理

自然界蚂蚁觅食行为 → 蚁群优化算法

- 自然界蚂蚁群体能够合作发现从蚁穴到食物源的最短路径。
- 通过对这种群体智能行为的抽象建模，研究者提出了蚁群优化算法（Ant Colony Optimization, ACO），为**最优化问题特别是组合优化问题的求解提供了一种强有力的手段。**
- 蚁群算法是一种仿生算法。

•蚁群算法的思想和起源

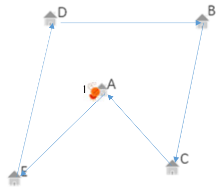
➢20 世纪90年代初，意大利学者Dorigo受蚂蚁觅食行为的启发，在其博士论文中首次提出了蚂蚁系统（Ant System）。

➢后来，Dorigo等人进一步将蚂蚁系统发展为一种通用的优化技术，称为蚁群优化算法（Ant Colony Optimization, ACO），简称为蚁群算法。

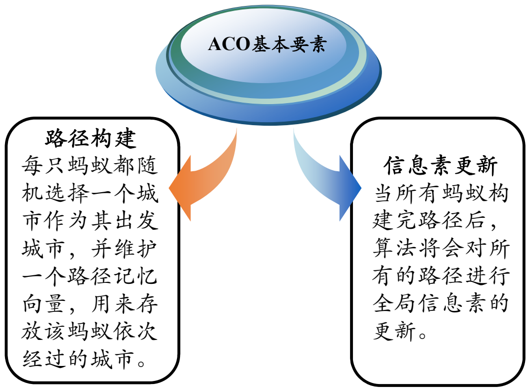
• 蚁群算法（ACO）是以旅行商（TSP）问题作为应用实例提出的。

旅行商（TSP）问题

旅行商问题，即TSP问题（Traveling Salesman Problem），又译为旅行推销员问题、货郎担问题，是数学领域中的著名问题之一。假设有一个旅行商人要拜访n个城市，他必须选择要走的路径，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。可选的路径方案有很多，而TSP问题的目标是希望选出所有路径之中路程最短的路径方案。



算法流程



• 路径构建：伪随机比例选择规则

设蚂蚁k当前所在城市为i，则其选择城市j作为下一个访问对象的概率：

$$p_k(i,j)=\begin{cases}\dfrac{[\tau(i,j)]^{\alpha}[\eta(i,j)]^{\beta}}{\sum\limits_{u\in J_k(i)}[\tau(i,u)]^{\alpha}[\eta(i,u)]^{\beta}}, & \text{if } j\in J_k(i)\\ 0, & \text{otherwise}\end{cases}$$

❖ 对于每只蚂蚁k，路径记忆向量R^k按照访问顺序记录了所有k已经经过的城市序号（禁忌表）。$J_k(i)$表示从城市i可以直接到达的、且又不在蚂蚁访问过的城市序列R^k中的城市集合。

❖ $\eta(i,j)$是一个启发式信息，通常由$\eta(i,j)=1/d_{ij}$直接计算。$\tau(i,j)$表示边(i,j)上的信息素量。

❖ 长度越短、信息素浓度越大的路径被蚂蚁选择的概率越大。α和β是两个预先设置的参数，用来控制信息素浓度与启发式信息的权重关系。

算法流程

- 信息素更新

❖ （1）在算法初始化时，问题空间中所有的边上的信息素都被初始化为τ_0。

❖ （2）算法每迭代一轮，问题空间中的所有路径上的信息素都会发生蒸发，所以为所有边上的信息素乘上一个小于1的常数。信息素蒸发是自然界本身固有的特征，在算法中能够帮助避免信息素的无限积累，使得算法可以快速丢弃之前构建过的较差的路径。

❖ （3）蚂蚁根据自己构建的路径长度在它们本轮经过的边上释放信息素。蚂蚁构建的路径越短、释放的信息素就越多。一条边被蚂蚁爬过的次数越多、它所获得的信息素也越多。

算法流程

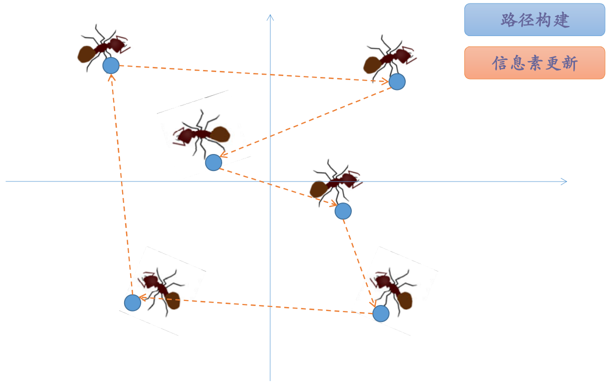
- 信息素更新

$$\tau(i,j) = (1-\rho)\cdot\tau(i,j) + \sum_{k=1}^{m}\Delta\tau_k(i,j),$$

$$\Delta\tau_k(i,j) = \begin{cases}(C_k)^{-1}, & \text{if } (i,j)\in R^k \\ 0, & \text{otherwise}\end{cases}$$

❖ **m是蚂蚁个数，ρ是信息素的蒸发率，规定$0<\rho\le 1$。** $\Delta\tau_k(i,j)$ **是第k只蚂蚁在它经过的边上释放的信息素量，它等于蚂蚁k本轮构建路径长度的倒数。C_k表示路径长度，它是R^k中所有边的长度和。**

算法流程



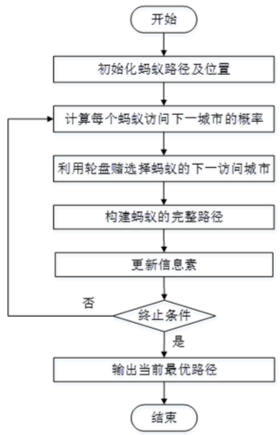
路径构建

信息素更新

算法流程

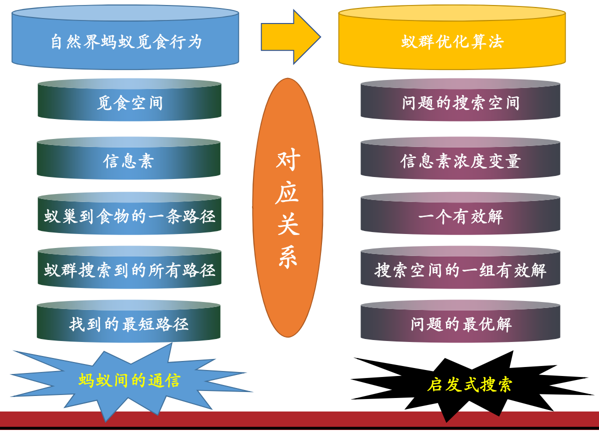
- 蚁群算法基本原理

算法：蚁群算法 (ACO)
Step 1：初始化参数，在初始时刻，设各城市连接路径的信息素浓度具有相同的值，m只蚂蚁放到n座城市；
Step 2：每只蚂蚁根据路径上的信息素和启发式信息，独立地访问下一座城市。确定从当前城市访问下一城市的概率；
Step 3：采用轮盘赌的方式，选择下一座城市。更新每只蚂蚁的禁忌表，直至所有蚂蚁遍历所有城市1次；
Step 4：更新每条路径上的信息素；
Step 5：若满足结束条件，即达到最大循环次数，则循环结束并输出程序计算结果，否则清空禁忌表并跳转到Step 2；



算法流程

- 蚁群算法基本原理



求解实例

- 例：给出用蚁群算法求解一个四城市的TSP问题的执行步骤，四个城市A、B、C、D之间的距离矩阵如下：

$$W = d_{ij} = \begin{bmatrix} \infty & 3 & 1 & 2 \\ 3 & \infty & 5 & 4 \\ 1 & 5 & \infty & 2 \\ 2 & 4 & 2 & \infty \end{bmatrix}$$

- 假设蚂蚁种群的规模**m=3**，参数 $\alpha = 1, \ \beta = 2, \ \rho = 0.5$

求解实例

步骤1：初始化。首先使用**贪心算法**得到路径（ACDBA），则 $C^{nn}=f(\mathrm{ACDBA})=1+2+4+3=10$。求得 $\tau_0=m/C^{nn}=3/10=0.3$。初始化所有边上的信息素 $\tau_{ij}=\tau_0$。

$$W = d_{ij} = \begin{bmatrix} \infty & 3 & 1 & 2 \\ 3 & \infty & 5 & 4 \\ 1 & 5 & \infty & 2 \\ 2 & 4 & 2 & \infty \end{bmatrix}$$

步骤2.1：为每只蚂蚁随机选择出发城市，假设蚂蚁1选择城市A，蚂蚁2选择城市B，蚂蚁3选择城市D。

求解实例

步骤2.2：为每只蚂蚁选择下一个城市。仅以蚂蚁1为例，当前城市 i=A，可访问城市集 $J_1(i)$={B,C,D}。计算蚂蚁1选择B、C、D作为下一访问城市的概率：

$$A \Rightarrow \begin{cases} B: \tau_{AB}^{\alpha} \times \eta_{AB}^{\beta} = 0.3^1 \times (1/3)^2 = 0.033 \\ C: \tau_{AC}^{\alpha} \times \eta_{AC}^{\beta} = 0.3^1 \times (1/1)^2 = 0.3 \\ D: \tau_{AD}^{\alpha} \times \eta_{AD}^{\beta} = 0.3^1 \times (1/2)^2 = 0.075 \end{cases}$$

$$p(B) = 0.033/(0.033+0.3+0.075) = 0.08$$

$$p(C) = 0.3/(0.033+0.3+0.075) = 0.74$$

$$p(D) = 0.075/(0.033+0.3+0.075) = 0.18$$

$$p_k(i,j) = \begin{cases} \dfrac{[\tau(i,j)]^{\alpha}[\eta(i,j)]^{\beta}}{\sum_{u \in J_k(i)} [\tau(i,u)]^{\alpha}[\eta(i,u)]^{\beta}}, & \text{if } j \in J_k(i) \\ 0, & \text{otherwise} \end{cases}$$

$$W = d_{ij} = \begin{bmatrix} \infty & 3 & 1 & 2 \\ 3 & \infty & 5 & 4 \\ 1 & 5 & \infty & 2 \\ 2 & 4 & 2 & \infty \end{bmatrix} \quad \alpha = 1, \ \beta = 2, \ \rho = 0.5$$

用轮盘赌法则选择下城市。假设产生的随机数 **q=random(0,1)=0.05**，则蚂蚁1将会选择城市B。

用同样的方法为蚂蚁2和3选择下一访问城市，则蚂蚁2选择城市D，蚂蚁3选择城市A。

基于之前的信息素计算p和更新选择下一条路径，如此重复，直到构建完整路径，得到C_k，才更新信息素

求解实例

步骤2.3： 当前蚂蚁1所在城市i=B，路径记忆向量R^1=(AB)，可访问城市集合$J_1(i)$={C,D}。计算蚂蚁1选择C、D作为下一城市的概率：

$$p_k(i,j)=\begin{cases}\dfrac{[\tau(i,j)]^\alpha[\eta(i,j)]^\beta}{\sum_{u\in J_k(i)}[\tau(i,u)]^\alpha[\eta(i,u)]^\beta}, & \text{if } j\in J_k(i)\\ 0, & \text{otherwise}\end{cases}$$

$$W=d_{ij}=\begin{bmatrix}\infty & 3 & 1 & 2\\ 3 & \infty & 5 & 4\\ 1 & 5 & \infty & 2\\ 2 & 4 & 2 & \infty\end{bmatrix}\quad \alpha=1,\ \beta=2,\ \rho=0.5$$

$$B\Rightarrow\begin{cases}C:\tau_{BC}^\alpha\times\eta_{BC}^\beta=0.3^1\times(1/5)^2=0.012\\ D:\tau_{BD}^\alpha\times\eta_{BD}^\beta=0.3^1\times(1/4)^2=0.019\end{cases}$$

$$p(C)=0.012/(0.012+0.019)=0.39$$

$$p(D)=0.019/(0.012+0.019)=0.61$$

用轮盘赌法则选择下城市。假设产生的随机数q=random(0,1)=0.67，则蚂蚁1将会选择城市D。

用同样的方法为蚂蚁2和3选择下一访问城市，则蚂蚁2选择城市C，蚂蚁3选择城市C。

求解实例

步骤2.4： 实际上此时路径已经构造完毕，蚂蚁1构建的路径为(ABDCA)。蚂蚁2构建的路径为(BDCAB)。蚂蚁3构建的路径为(DACBD)。

步骤3： 信息素更新。计算每只蚂蚁构建的路径长度：

C_1=3+4+2+1=10，C_2=4+2+1+3=10，C_3=2+1+5+4=12，

更新每条边上的信息素：

$$\tau(i,j)=(1-\rho)\cdot\tau(i,j)+\sum_{k=1}^{m}\Delta\tau_k(i,j),$$

$$\Delta\tau_k(i,j)=\begin{cases}(C_k)^{-1}, & if\ (i,j)\in R^k\\ 0, & \text{otherwise}\end{cases}.$$

$$\alpha=1,\ \beta=2,\ \rho=0.5$$

$$\tau_{AB}=(1-\rho)\times\tau_{AB}+\sum_{k=1}^{3}\Delta\tau_{AB}^k=0.5\times0.3+(1/10+1/10)=0.35$$

$$\tau_{AC}=(1-\rho)\times\tau_{AC}+\sum_{k=1}^{3}\Delta\tau_{AC}^k=0.5\times0.3+(\frac{1}{10}+\frac{1}{10}+\frac{1}{12})=0.43$$

如此，依次计算出问题空间内所有边更新后的信息素量。

$$W=d_{ij}=\begin{bmatrix}\infty & 3 & 1 & 2\\ 3 & \infty & 5 & 4\\ 1 & 5 & \infty & 2\\ 2 & 4 & 2 & \infty\end{bmatrix}$$

注意：AC边的信息素计算需要计算蚂蚁1、蚂蚁2和蚂蚁3的累积信息素。

求解实例

步骤4： 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2.1继续执行。

$$p_k(i,j)=\begin{cases}\dfrac{[\tau(i,j)]^\alpha[\eta(i,j)]^\beta}{\sum_{u\in J_k(i)}[\tau(i,u)]^\alpha[\eta(i,u)]^\beta}, & \text{if } j\in J_k(i)\\ 0, & \text{otherwise}\end{cases}\qquad \tau(i,j)=(1-\rho)\cdot\tau(i,j)+\sum_{k=1}^{m}\Delta\tau_k(i,j),$$

α：α过大，蚂蚁选择以前走过的路径的可能性越大，搜索的随机性就会减弱。

β：β过大，蚂蚁在某个局部点上选择局部最短路径的可能性就越大，虽然这个时候算法的收敛速度得以加快，但搜索易陷入局部最优。

ρ：ρ过小以前搜索过的路径被再次选择的可能性过大，搜索的随机性就会减弱。

蚁群算法优缺点

算法优点

- 蚁群算法与其他启发式算法相比，在求解性能上，具有很强的鲁棒性，搜索能力较强。
- 蚁群算法是一种基于种群的算法，具有本质并行性，易于并行实现。
- 蚁群算法很容易与其他算法如遗传算法、粒子群算法结合，以改善算法性能。

算法不足

- 如果初始化参数设置不当，会导致求解速度很慢且所得解的质量特别差。
- 基本蚁群算法即无改进的蚁群算法，计算量大，求解所需时间较长。

小结

蚁群算法的算法流程

蚁群算法的基本要素：

- **路径构建**
- **信息素更新**

思考

- 禁忌表在路径构建过程中起到什么作用呢?
- 对于相同的问题多次运行蚁群算法，得到的结果可能不一样，原因是什么?

工件排序

设有n个工件等待在一台机床上加工，加工完i，接着加工j，这中间机器需要花费一定的准备时间t_{ij}，问如何安排加工顺序使总调整时间最短?

此问题可用TSP的方法求解，n个工件对应n个顶点，t_{ij}表示边(i,j)上的权重。