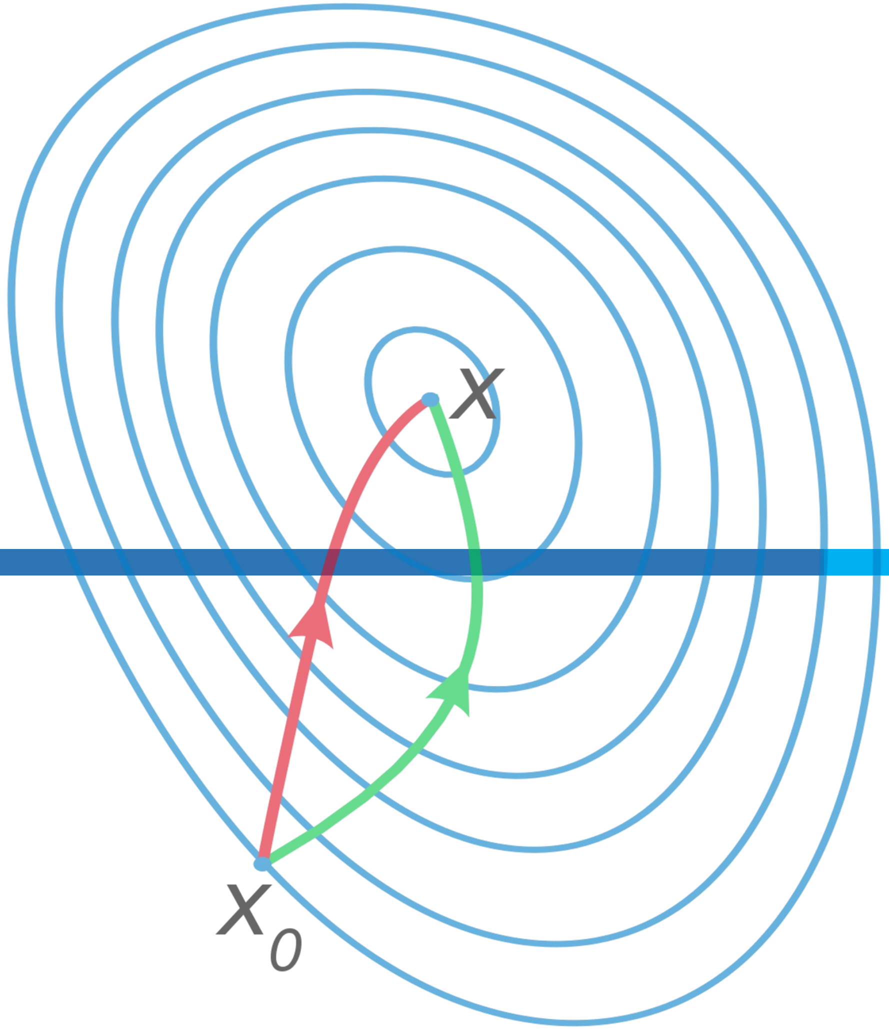
最优化方法



第七周

计算机学院
余皓然

2024/4/8

进化算法之差分进化算法

Part1 如何从复杂度的角度衡量算法好坏？

Part2 针对一些有特定形式的最优化问题，如何找到最优解？

Part3 针对较复杂的最优化问题（如NP难问题），如何找到较好解？

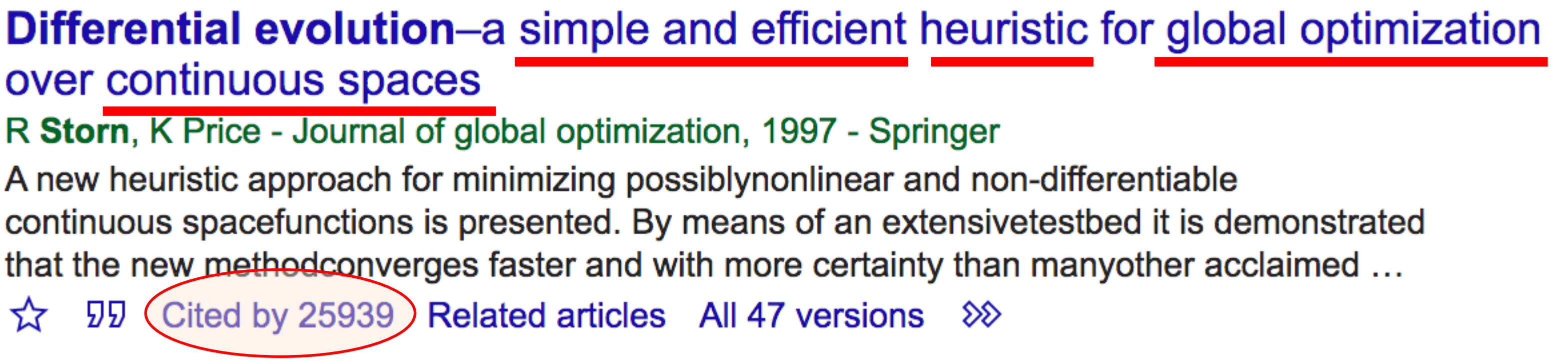
局部搜索、模拟退火、遗传算法、差分进化算法、蚁群算法、粒子群优化算法、人工蜂群算法、贝叶斯优化

Part4 针对较复杂的多目标最优化问题，如何找到较好解？

基本介绍

差分进化（Differential Evolution， DE）

主要用于解决连续决策变量的问题

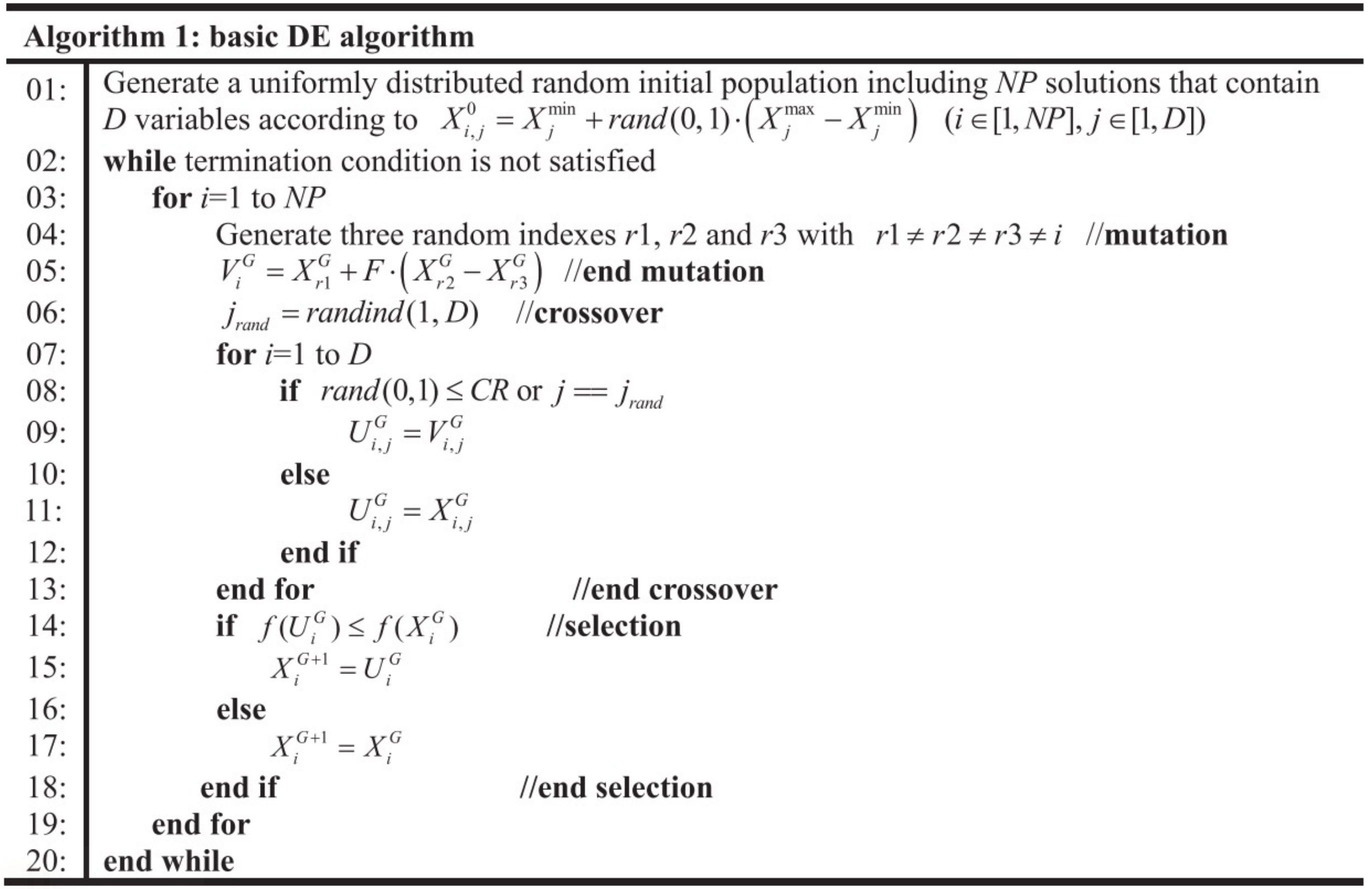


Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces

R **Storn**, K Price - Journal of global optimization, 1997 - Springer

A new heuristic approach for minimizing possiblynonlinear and non-differentiable continuous spacefunctions is presented. By means of an extensivetestbed it is demonstrated that the new methodconverges faster and with more certainty than manyother acclaimed …

差分进化



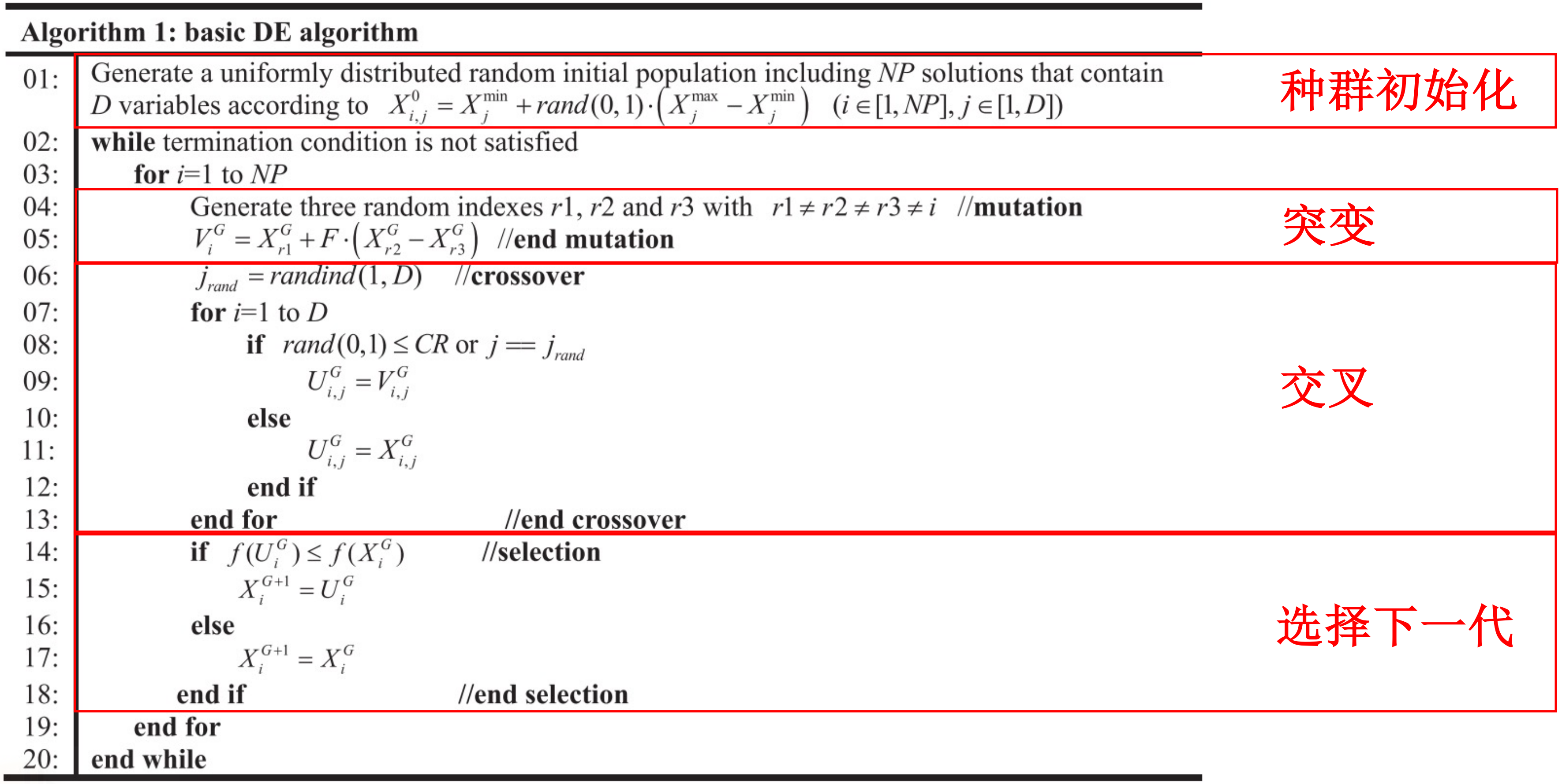
Algorithm 1: basic DE algorithm

```
Generate a uniformly distributed random initial population including $NP$ solutions that contain
$D$ variables according to $X_{i,j}^{0} = X_{j}^{\min} + rand(0,1)\cdot\left(X_{j}^{\max} - X_{j}^{\min}\right)$ $(i \in [1, NP], j \in [1, D])$
while termination condition is not satisfied
    for $i$=1 to $NP$
        Generate three random indexes $r1$, $r2$ and $r3$ with $r1 \neq r2 \neq r3 \neq i$   //mutation
        $V_{i}^{G} = X_{r1}^{G} + F\cdot\left(X_{r2}^{G} - X_{r3}^{G}\right)$   //end mutation
        $j_{rand} = randind(1, D)$   //crossover
        for $i$=1 to $D$
            if $rand(0,1) \leq CR$ or $j == j_{rand}$
                $U_{i,j}^{G} = V_{i,j}^{G}$
            else
                $U_{i,j}^{G} = X_{i,j}^{G}$
            end if
        end for                        //end crossover
        if $f(U_{i}^{G}) \leq f(X_{i}^{G})$        //selection
            $X_{i}^{G+1} = U_{i}^{G}$
        else
            $X_{i}^{G+1} = X_{i}^{G}$
        end if                         //end selection
    end for
end while
```

图片取自Li et al., “A novel hybrid differential evolution algorithm with modified CoDE and JADE”

差分进化

突变 和 交叉 的顺序和遗传算法相反



Algorithm 1: basic DE algorithm

```
Generate a uniformly distributed random initial population including NP solutions that contain
D variables according to $X_{i,j}^{0} = X_{j}^{\min} + rand(0,1)\cdot\left(X_{j}^{\max} - X_{j}^{\min}\right)$  $(i \in [1, NP], j \in [1, D])$
while termination condition is not satisfied
    for $i$=1 to $NP$
        Generate three random indexes $r1$, $r2$ and $r3$ with $r1 \neq r2 \neq r3 \neq i$  //mutation
        $V_{i}^{G} = X_{r1}^{G} + F\cdot\left(X_{r2}^{G} - X_{r3}^{G}\right)$  //end mutation
        $j_{rand} = randind(1, D)$  //crossover
        for $i$=1 to $D$
            if $rand(0,1) \leq CR$ or $j == j_{rand}$
                $U_{i,j}^{G} = V_{i,j}^{G}$
            else
                $U_{i,j}^{G} = X_{i,j}^{G}$
            end if
        end for                    //end crossover
        if $f(U_{i}^{G}) \leq f(X_{i}^{G})$          //selection
            $X_{i}^{G+1} = U_{i}^{G}$
        else
            $X_{i}^{G+1} = X_{i}^{G}$
        end if                     //end selection
    end for
end while
```

种群初始化

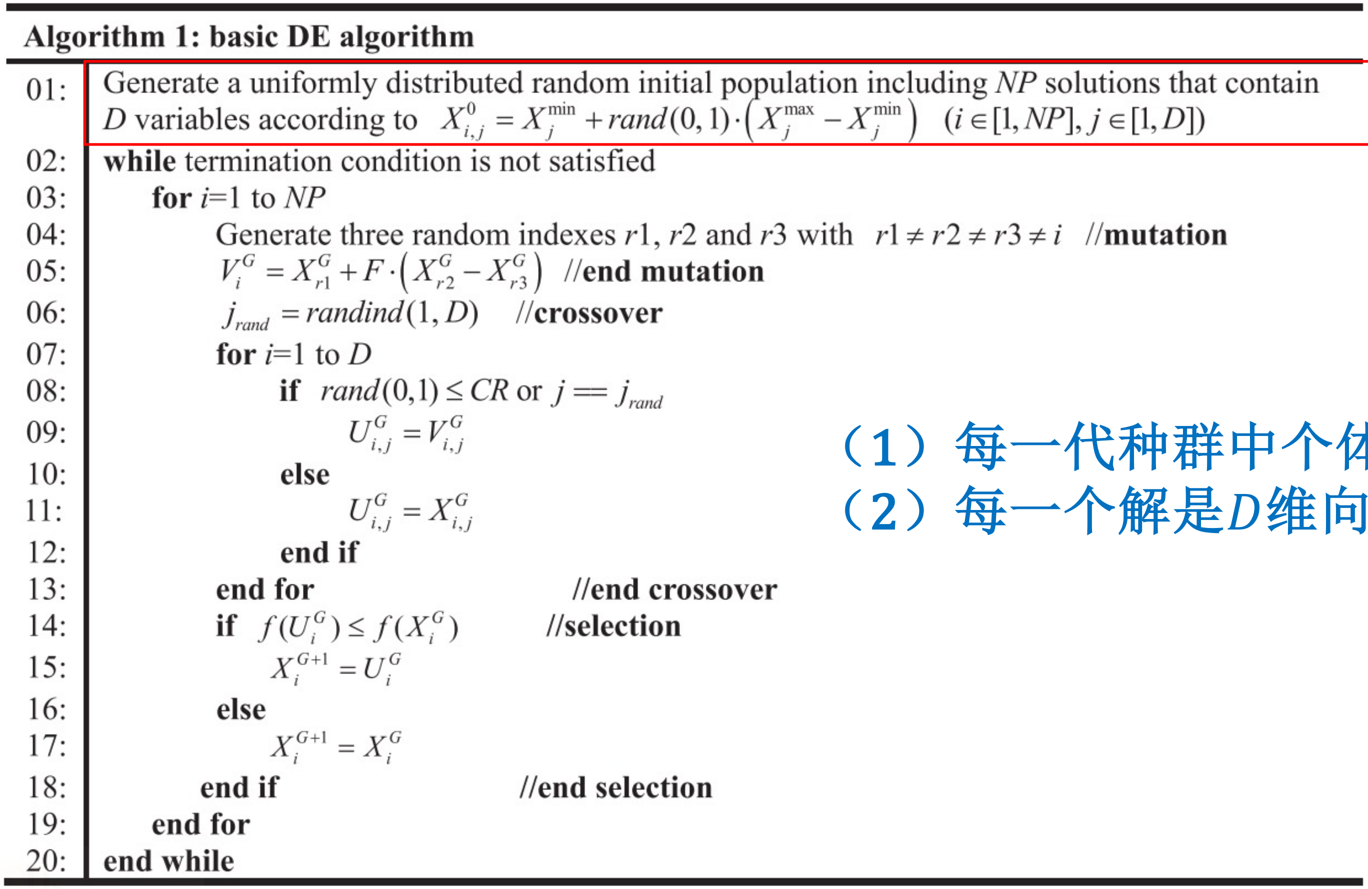
突变

交叉

选择下一代

图片取自Li et al., “A novel hybrid differential evolution algorithm with modified CoDE and JADE”

差分进化



Algorithm 1: basic DE algorithm

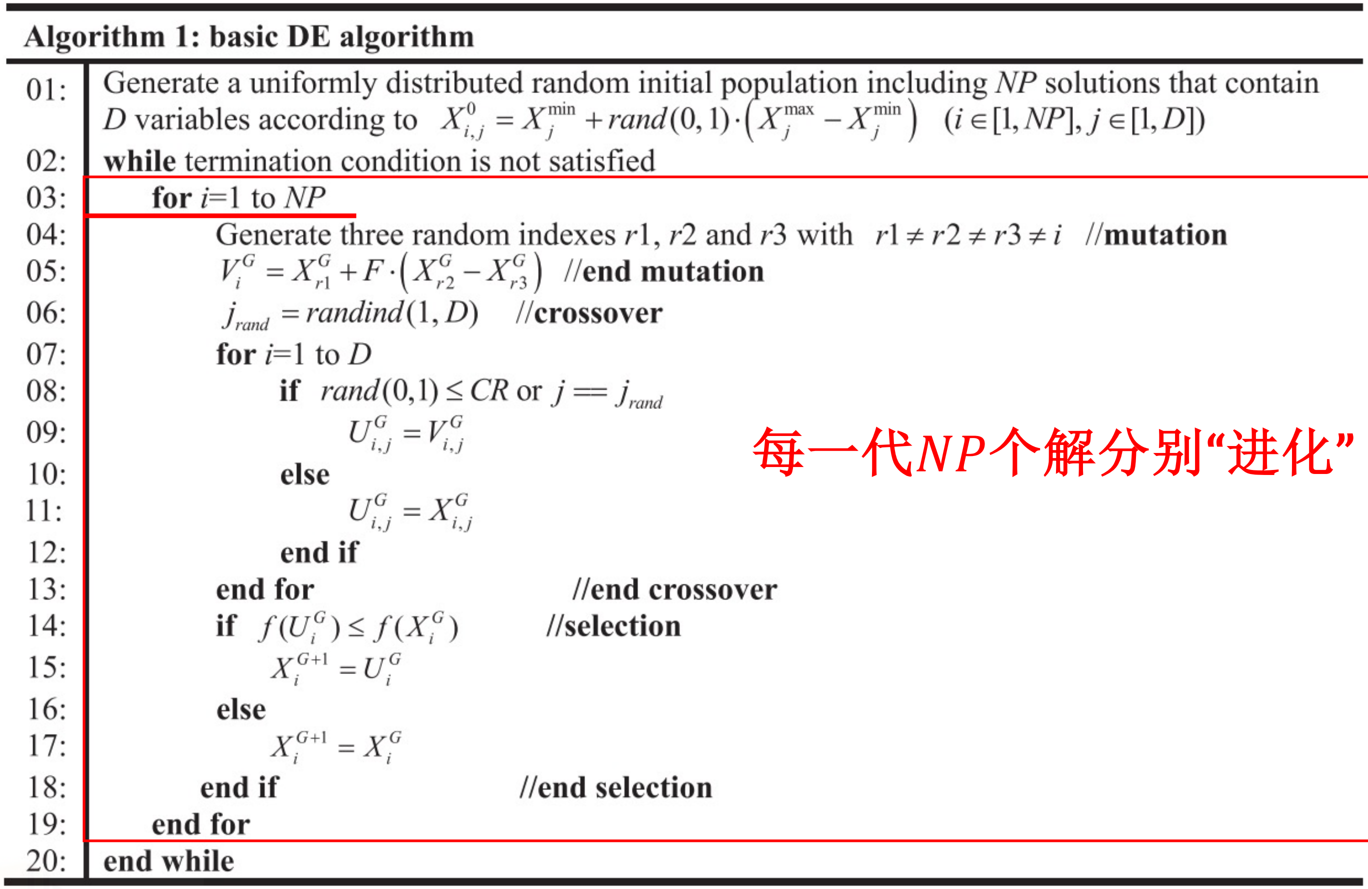
```
Generate a uniformly distributed random initial population including $NP$ solutions that contain
$D$ variables according to $X_{i,j}^{0} = X_{j}^{\min} + rand(0,1) \cdot \left(X_{j}^{\max} - X_{j}^{\min}\right) \quad (i \in [1, NP], j \in [1, D])$
while termination condition is not satisfied
    for $i$=1 to $NP$
        Generate three random indexes $r1$, $r2$ and $r3$ with $r1 \neq r2 \neq r3 \neq i$   //mutation
        $V_{i}^{G} = X_{r1}^{G} + F \cdot \left(X_{r2}^{G} - X_{r3}^{G}\right)$   //end mutation
        $j_{rand} = randind(1, D)$   //crossover
        for $i$=1 to $D$
            if $rand(0,1) \leq CR$ or $j == j_{rand}$
                $U_{i,j}^{G} = V_{i,j}^{G}$
            else
                $U_{i,j}^{G} = X_{i,j}^{G}$
            end if
        end for                    //end crossover
        if $f(U_{i}^{G}) \leq f(X_{i}^{G})$        //selection
            $X_{i}^{G+1} = U_{i}^{G}$
        else
            $X_{i}^{G+1} = X_{i}^{G}$
       end if                    //end selection
    end for
end while
```

种群初始化

（1）每一代种群中个体数目确定，即*NP*

（2）每一个解是*D*维向量，包含*D*个元素

差分进化



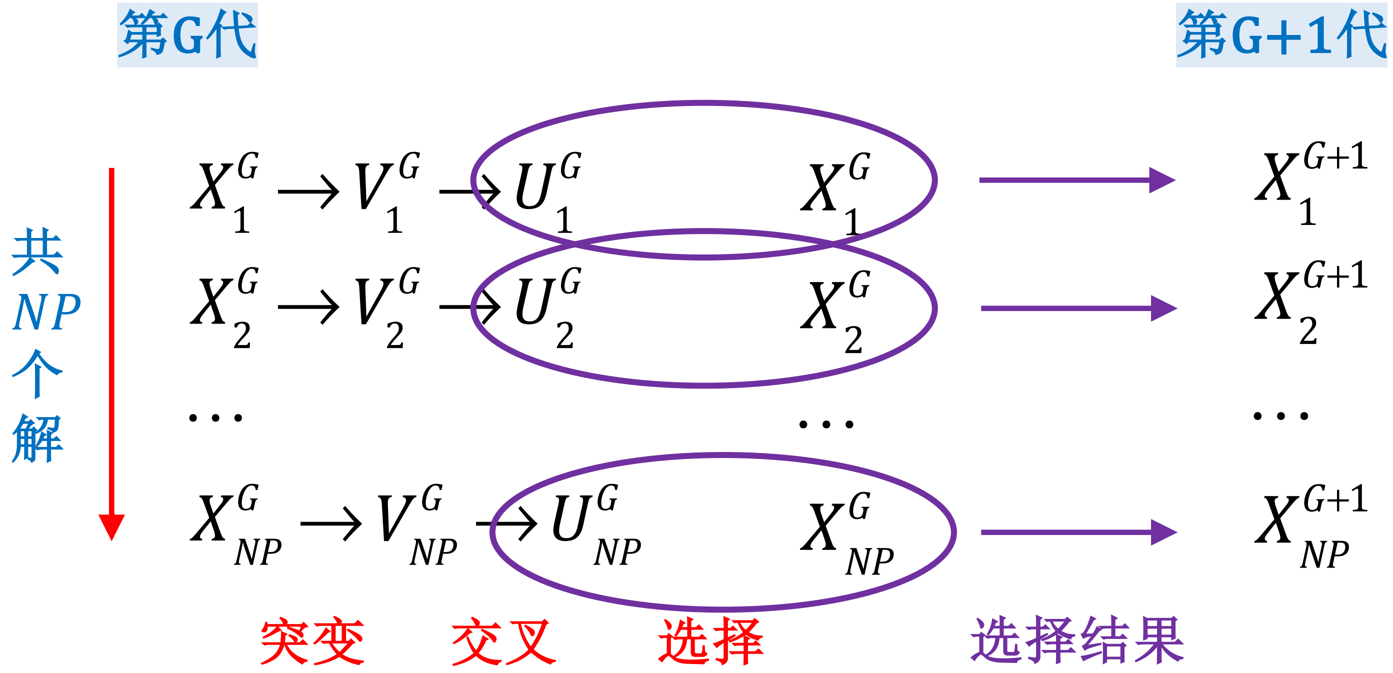
Algorithm 1: basic DE algorithm

```
Generate a uniformly distributed random initial population including $NP$ solutions that contain $D$ variables according to $X_{i,j}^{0} = X_{j}^{\min} + rand(0,1)\cdot\left(X_{j}^{\max} - X_{j}^{\min}\right)$ $(i \in [1, NP], j \in [1, D])$
while termination condition is not satisfied
    for $i$=1 to $NP$
        Generate three random indexes $r1$, $r2$ and $r3$ with $r1 \neq r2 \neq r3 \neq i$   //mutation
        $V_{i}^{G} = X_{r1}^{G} + F\cdot\left(X_{r2}^{G} - X_{r3}^{G}\right)$   //end mutation
        $j_{rand} = randind(1, D)$   //crossover
        for $i$=1 to $D$
            if $rand(0,1) \leq CR$ or $j == j_{rand}$
                $U_{i,j}^{G} = V_{i,j}^{G}$
            else
                $U_{i,j}^{G} = X_{i,j}^{G}$
            end if
        end for                         //end crossover
        if $f(U_{i}^{G}) \leq f(X_{i}^{G})$   //selection
            $X_{i}^{G+1} = U_{i}^{G}$
        else
            $X_{i}^{G+1} = X_{i}^{G}$
        end if                          //end selection
    end for
end while
```

每一代NP个解分别“进化”

差分进化

每一代NP个解分别“进化”



第G代

第G+1代

共NP个解

$X_1^G \rightarrow V_1^G \rightarrow U_1^G \quad X_1^G \longrightarrow X_1^{G+1}$

$X_2^G \rightarrow V_2^G \rightarrow U_2^G \quad X_2^G \longrightarrow X_2^{G+1}$

$\cdots \qquad \cdots \qquad \cdots$

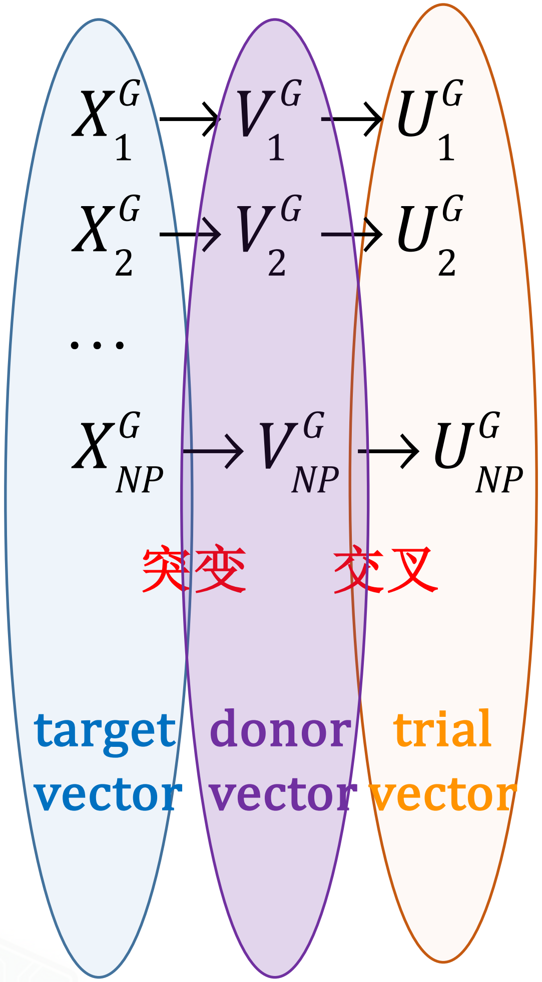
$X_{NP}^G \rightarrow V_{NP}^G \rightarrow U_{NP}^G \quad X_{NP}^G \longrightarrow X_{NP}^{G+1}$

突变　交叉　选择　选择结果

注意以上都是向量表达

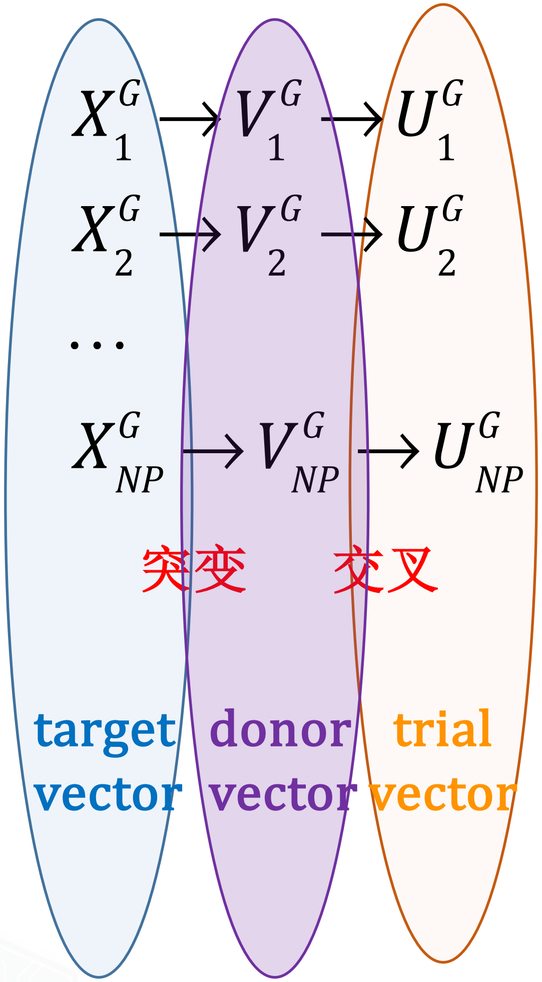
在遗传算法中，繁殖（基因交叉）是以对为单位进行

差分进化

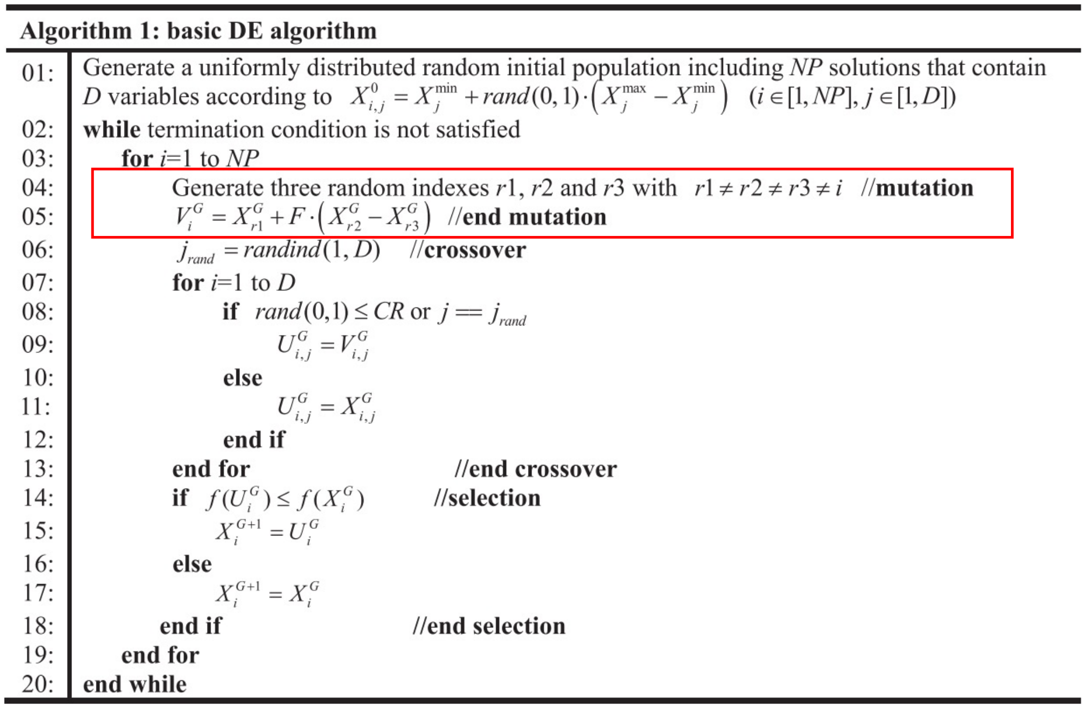

$X_1^G \rightarrow V_1^G \rightarrow U_1^G$
$X_2^G \rightarrow V_2^G \rightarrow U_2^G$
$\cdots$
$X_{NP}^G \rightarrow V_{NP}^G \rightarrow U_{NP}^G$
突变
交叉
target vector
donor vector
trial vector

突变:
为靶向量(target vector)确定
捐赠向量(donor vector)

差分进化


$X_1^G \rightarrow V_1^G \rightarrow U_1^G$
$X_2^G \rightarrow V_2^G \rightarrow U_2^G$
…
$X_{NP}^G \rightarrow V_{NP}^G \rightarrow U_{NP}^G$
突变
交叉
target vector
donor vector
trial vector

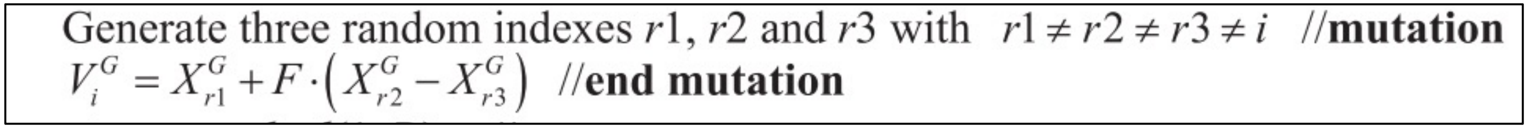
突变:
为靶向量(target vector)确定
捐赠向量(donor vector)



Algorithm 1: basic DE algorithm

```
Generate a uniformly distributed random initial population including $NP$ solutions that contain
$D$ variables according to $X_{i,j}^0 = X_j^{\min} + rand(0,1)\cdot\left(X_j^{\max} - X_j^{\min}\right)$ $(i \in [1, NP], j \in [1, D])$
while termination condition is not satisfied
    for $i$=1 to $NP$
        Generate three random indexes $r1$, $r2$ and $r3$ with $r1 \neq r2 \neq r3 \neq i$ //mutation
        $V_i^G = X_{r1}^G + F\cdot\left(X_{r2}^G - X_{r3}^G\right)$ //end mutation
        $j_{rand} = randind(1, D)$ //crossover
        for $i$=1 to $D$
            if $rand(0,1) \leq CR$ or $j == j_{rand}$
                $U_{i,j}^G = V_{i,j}^G$
            else
                $U_{i,j}^G = X_{i,j}^G$
            end if
        end for                    //end crossover
        if $f(U_i^G) \leq f(X_i^G)$          //selection
            $X_i^{G+1} = U_i^G$
        else
            $X_i^{G+1} = X_i^G$
       end if                    //end selection
    end for
end while
```

突变



```
Generate three random indexes $r1$, $r2$ and $r3$ with $r1 \neq r2 \neq r3 \neq i$   //mutation
$V_i^G = X_{r1}^G + F \cdot \left( X_{r2}^G - X_{r3}^G \right)$   //end mutation
```

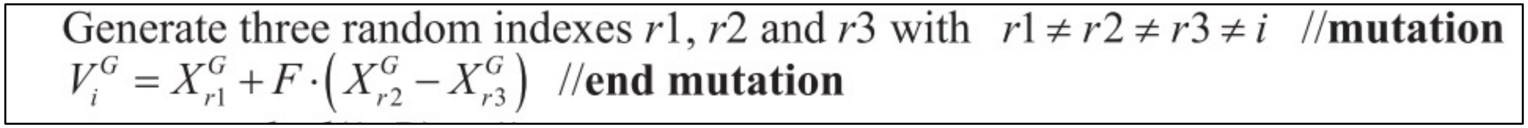
比如为$i = 1$靶向量找捐赠向量

$$\underset{\text{不能选}}{X_1^G}, \underbrace{X_2^G, X_3^G, X_4^G, \ldots, X_{NP}^G}_{\text{选三个不同的向量}}$$

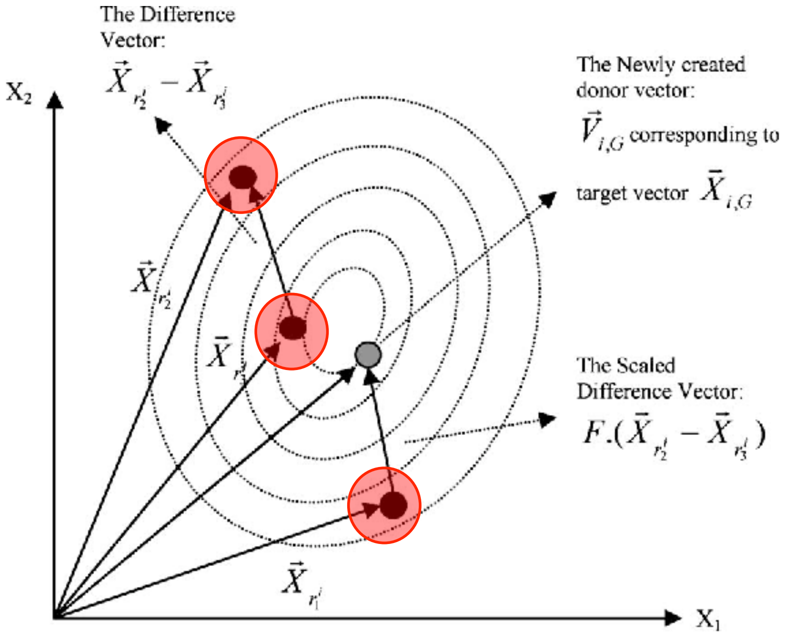
$F \in [0,2]$是自定义的系数

差分进化算法需要NP至少为4

突变



```
Generate three random indexes $r1$, $r2$ and $r3$ with  $r1 \neq r2 \neq r3 \neq i$  //mutation
$V_i^G = X_{r1}^G + F \cdot \left( X_{r2}^G - X_{r3}^G \right)$  //end mutation
```

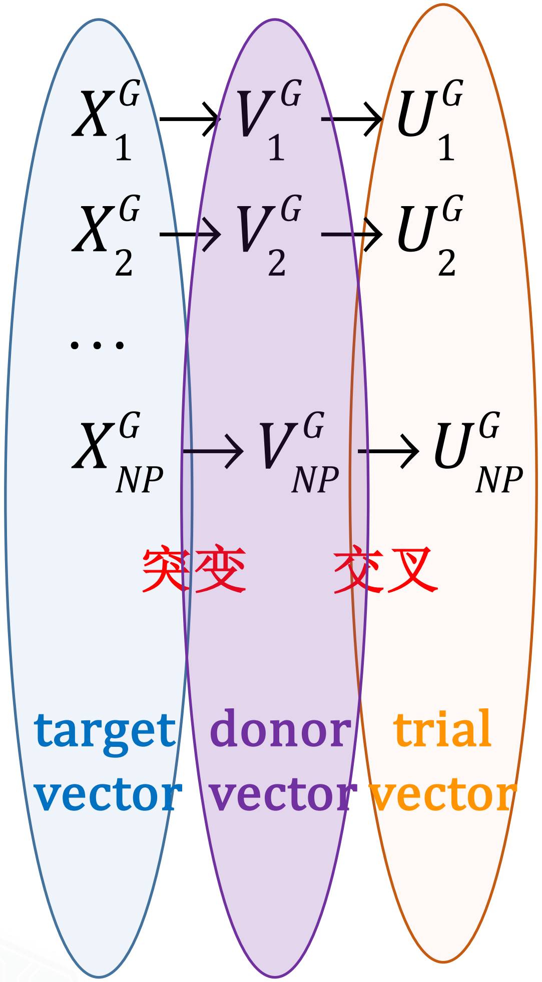


The Difference Vector:
$\vec{X}_{r_2^i} - \vec{X}_{r_3^i}$
X_2
The Newly created donor vector:
$\vec{V}_{i,G}$ corresponding to
target vector $\vec{X}_{i,G}$
$\vec{X}_{r_2^i}$
$\vec{X}_{r_3^i}$
The Scaled Difference Vector:
$F.(\vec{X}_{r_2^i} - \vec{X}_{r_3^i})$
$\vec{X}_{r_1^i}$
X_1

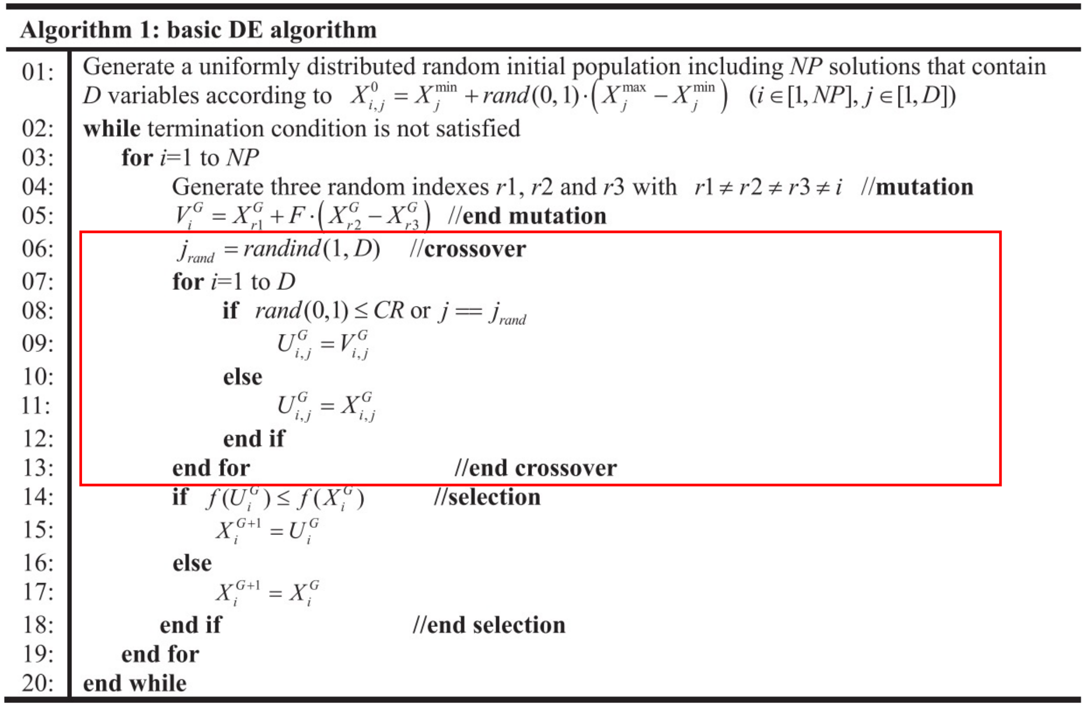
g. 2. Illustrating a simple DE mutation scheme in 2-D parametric space.

图片取自Swagatam Das, P. Suganthan , “Differential Evolution: A Survey of the State-of-the-Art”

差分进化


$X_1^G \rightarrow V_1^G \rightarrow U_1^G$
$X_2^G \rightarrow V_2^G \rightarrow U_2^G$
…
$X_{NP}^G \rightarrow V_{NP}^G \rightarrow U_{NP}^G$
突变
交叉
target vector
donor vector
trial vector

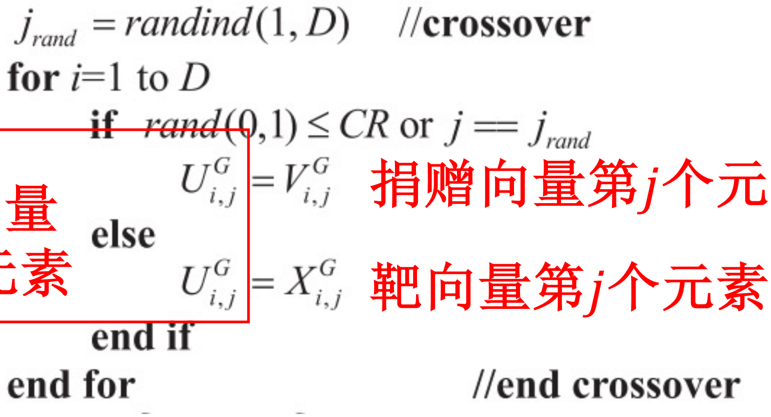
交叉: 根据靶向量(target vector)和捐赠向量(donor vector)确定试验向量(trial vector)



Algorithm 1: basic DE algorithm

```
Generate a uniformly distributed random initial population including NP solutions that contain
D variables according to $X_{i,j}^0 = X_j^{\min} + rand(0,1) \cdot \left(X_j^{\max} - X_j^{\min}\right) \quad (i \in [1, NP], j \in [1, D])$
while termination condition is not satisfied
    for i=1 to NP
        Generate three random indexes r1, r2 and r3 with $r1 \neq r2 \neq r3 \neq i$ //mutation
        $V_i^G = X_{r1}^G + F \cdot \left(X_{r2}^G - X_{r3}^G\right)$ //end mutation
        $j_{rand} = randind(1, D)$ //crossover
        for i=1 to D
            if $rand(0,1) \leq CR$ or $j == j_{rand}$
                $U_{i,j}^G = V_{i,j}^G$
            else
                $U_{i,j}^G = X_{i,j}^G$
            end if
        end for                          //end crossover
        if $f(U_i^G) \leq f(X_i^G)$          //selection
            $X_i^{G+1} = U_i^G$
        else
            $X_i^{G+1} = X_i^G$
        end if                           //end selection
    end for
end while
```

差分进化



```
$j_{rand} = randind(1, D)$    //crossover
for $i$=1 to $D$
    if $rand(0,1) \le CR$ or $j == j_{rand}$
        $U_{i,j}^{G} = V_{i,j}^{G}$
    else
        $U_{i,j}^{G} = X_{i,j}^{G}$
    end if
end for                //end crossover
```

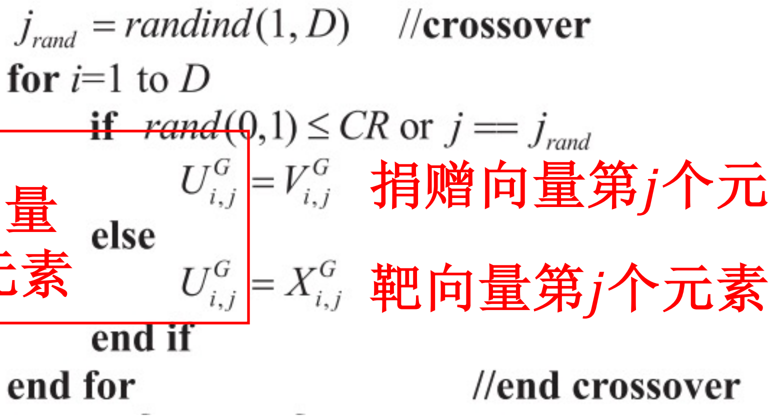
试验向量第j个元素

捐赠向量第j个元素

靶向量第j个元素

确定 从捐赠向量取值的概率CR以及j_{rand}

差分进化



```
$j_{rand} = randind(1, D)$    //crossover
for $i$=1 to $D$
    if  $rand(0,1) \le CR$ or $j == j_{rand}$
        $U_{i,j}^{G} = V_{i,j}^{G}$
    else
        $U_{i,j}^{G} = X_{i,j}^{G}$
    end if
end for                       //end crossover
```

试验向量第j个元素

捐赠向量第j个元素

靶向量第j个元素

确定 从捐赠向量取值的概率CR以及j_{rand}

靶向量
2.2
3.1
0.4
2.1

捐赠向量
0.5
2.1
3.5
4.1

比如取$CR = 0.7$

$j_{rand} = 2$

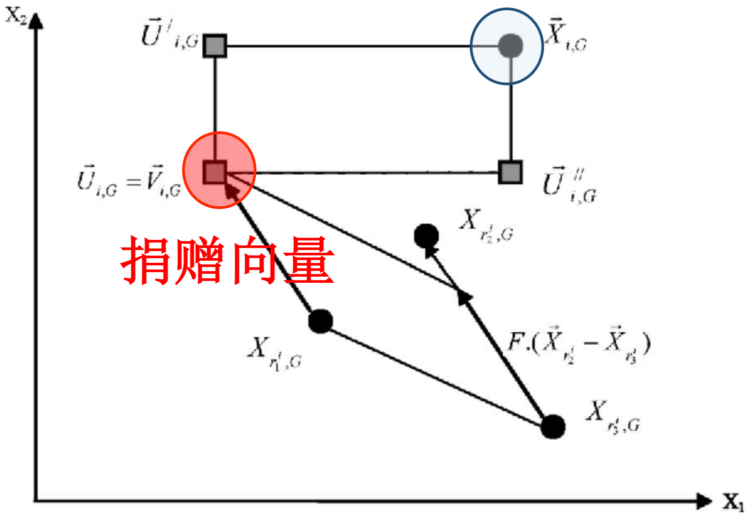
试验向量
2.2
2.1
3.5
4.1

j_{rand}的目的是确保至少一个元素取自捐赠向量，及确保试验向量和靶向量不同

差分进化

```
$j_{rand} = randind(1, D)$    //crossover
for $i$=1 to $D$
    if  $rand(0,1) \leq CR$ or $j == j_{rand}$
        $U_{i,j}^{G} = V_{i,j}^{G}$   捐赠向量第$j$个元素
    else
        $U_{i,j}^{G} = X_{i,j}^{G}$   靶向量第$j$个元素
    end if
end for                    //end crossover
```

靶向量

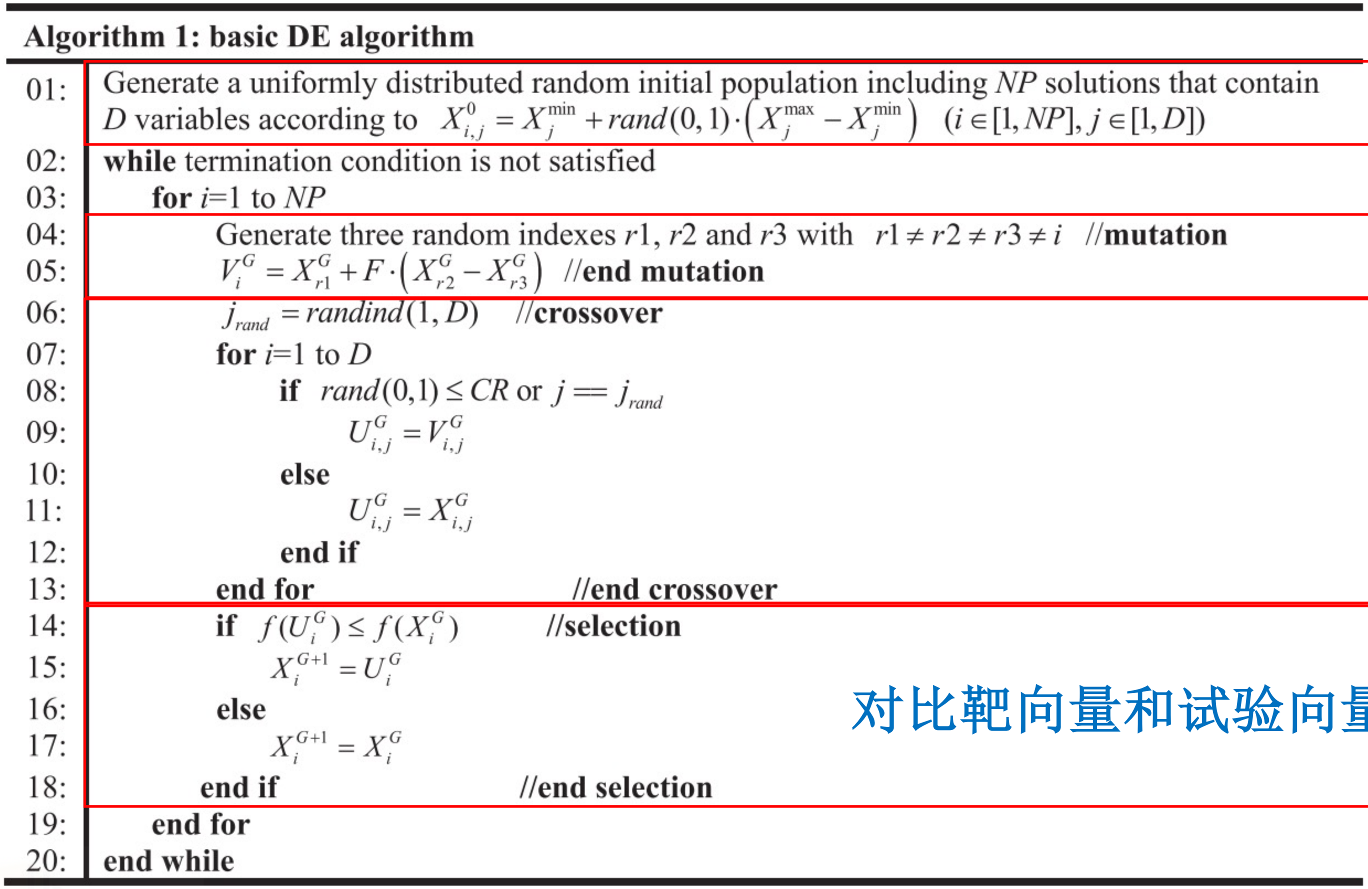


捐赠向量

Fig. 3. Different possible trial vectors formed due to uniform/binomial crossover between the target and the mutant vectors in 2-D search space.

图片取自Swagatam Das, P. Suganthan , “Differential Evolution: A Survey of the State-of-the-Art”

差分进化



Algorithm 1: basic DE algorithm

```
Generate a uniformly distributed random initial population including $NP$ solutions that contain
$D$ variables according to $X_{i,j}^{0} = X_j^{\min} + rand(0,1)\cdot\left(X_j^{\max} - X_j^{\min}\right)$ $(i \in [1, NP], j \in [1, D])$
while termination condition is not satisfied
    for $i$=1 to $NP$
        Generate three random indexes $r1$, $r2$ and $r3$ with $r1 \neq r2 \neq r3 \neq i$ //mutation
        $V_i^G = X_{r1}^G + F\cdot\left(X_{r2}^G - X_{r3}^G\right)$ //end mutation
        $j_{rand} = randind(1, D)$ //crossover
        for $i$=1 to $D$
            if $rand(0,1) \leq CR$ or $j == j_{rand}$
                $U_{i,j}^G = V_{i,j}^G$
            else
                $U_{i,j}^G = X_{i,j}^G$
            end if
        end for //end crossover
        if $f(U_i^G) \leq f(X_i^G)$ //selection
            $X_i^{G+1} = U_i^G$
        else
            $X_i^{G+1} = X_i^G$
        end if //end selection
    end for
end while
```

种群初始化

突变捐赠向量

交叉试验向量

对比靶向量和试验向量 选择下一代

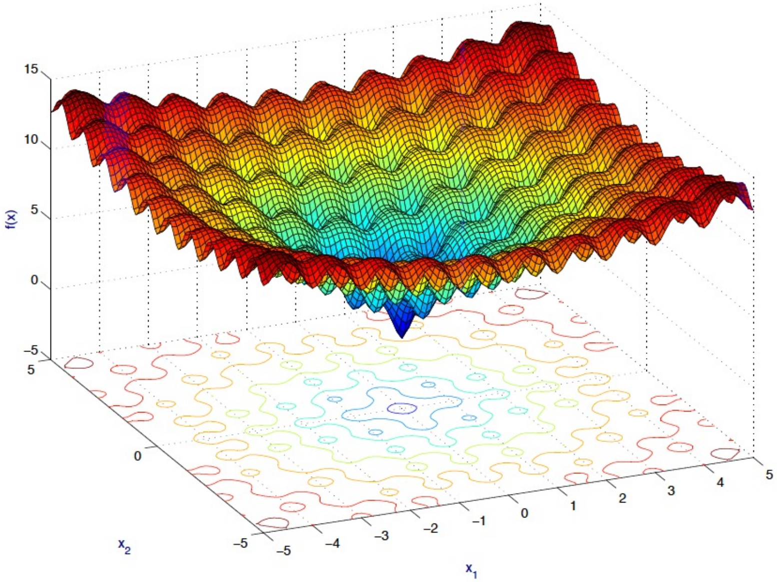
例子

Ackley’s function，用于测试解决非凸优化问题解法性能的函数

- DE with $N = 10$, $F = 0.5$ and $CR = 0.1$
- Ackley’s function

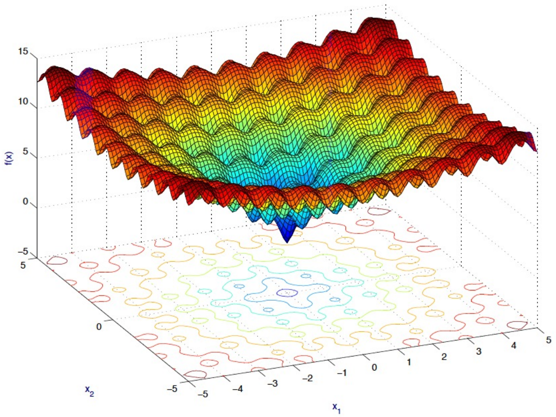
$$f(x_1, x_2) = 20+e-20\exp\left(-0.2\sqrt{\frac{1}{n}(x_1^2 + x_2^2)}\right)-\exp\left(\frac{1}{n}\left(\cos(2\pi x_1) + \cos(2\pi x_2)\right)\right)$$

- Find $x^* \in [-5, 5]$ such that $f(x^*) \leq f(x)\ \forall x \in [-5, 5]$
- $f(x^*) = 0;\ x^* = (0, 0)$

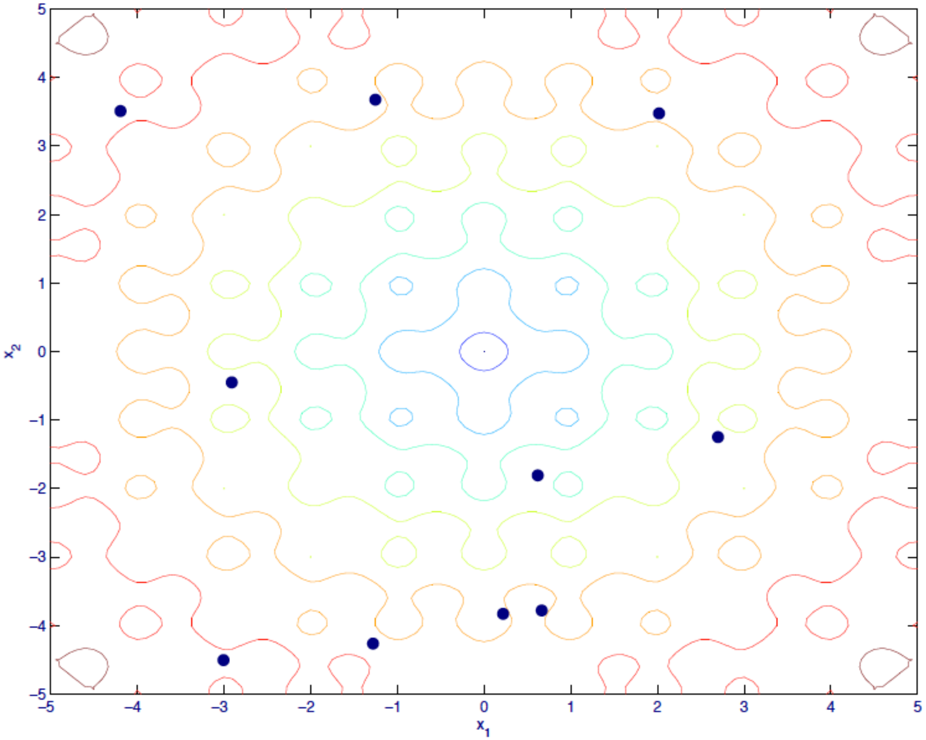


内容取自Kelly Fleetwood ”An Introduction to Differential Evolution”

例子

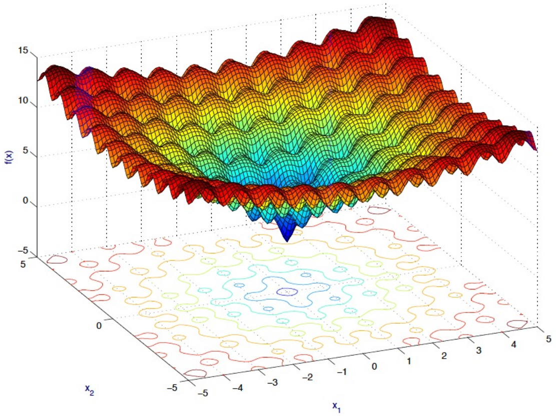


种群规模:10

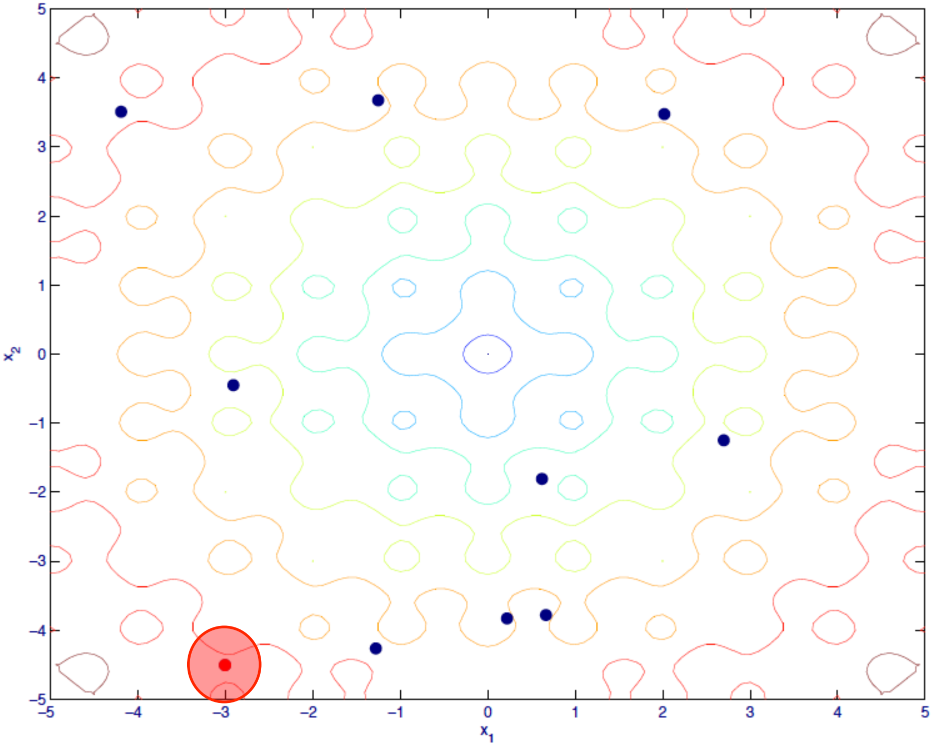


内容取自Kelly Fleetwood ”An Introduction to Differential Evolution”

例子



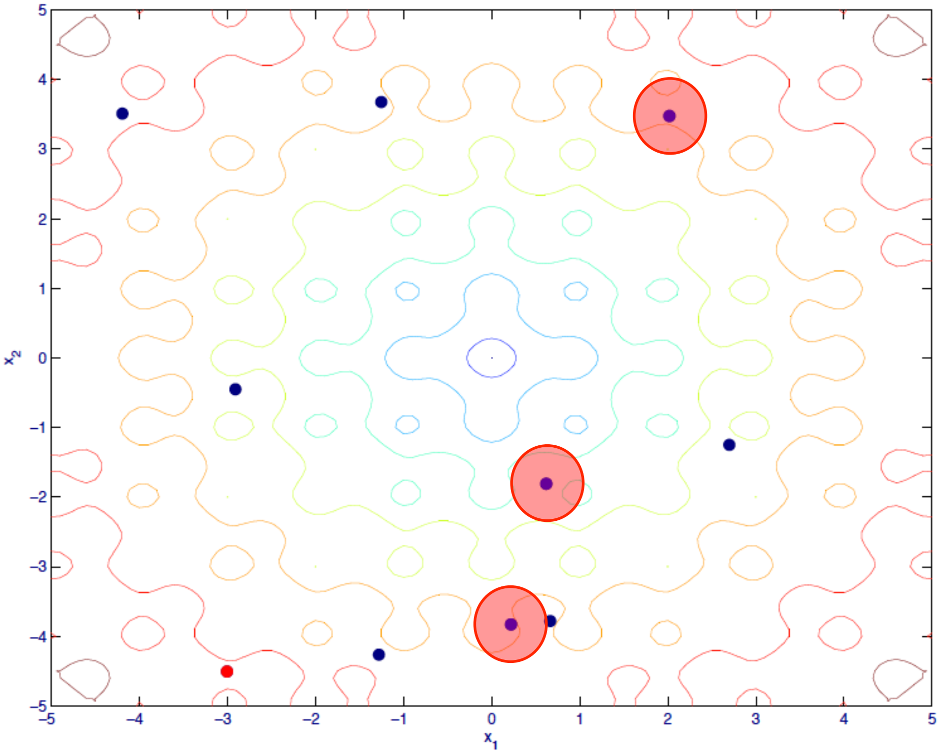
选择靶向量



内容取自Kelly Fleetwood ”An Introduction to Differential Evolution”

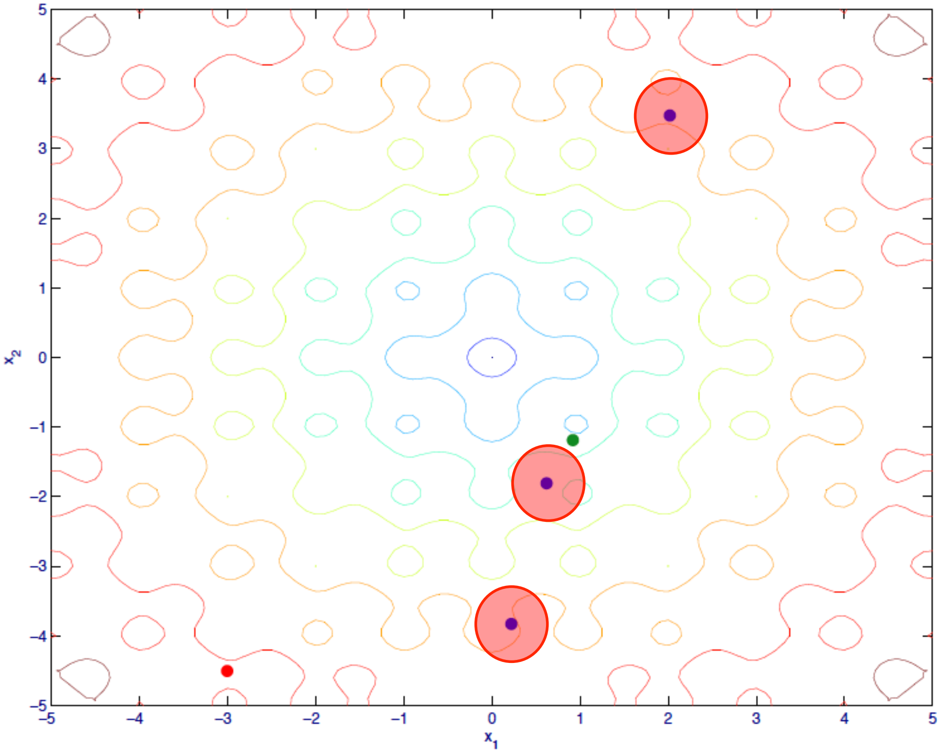
例子

选择用于构建捐赠向量的三个向量



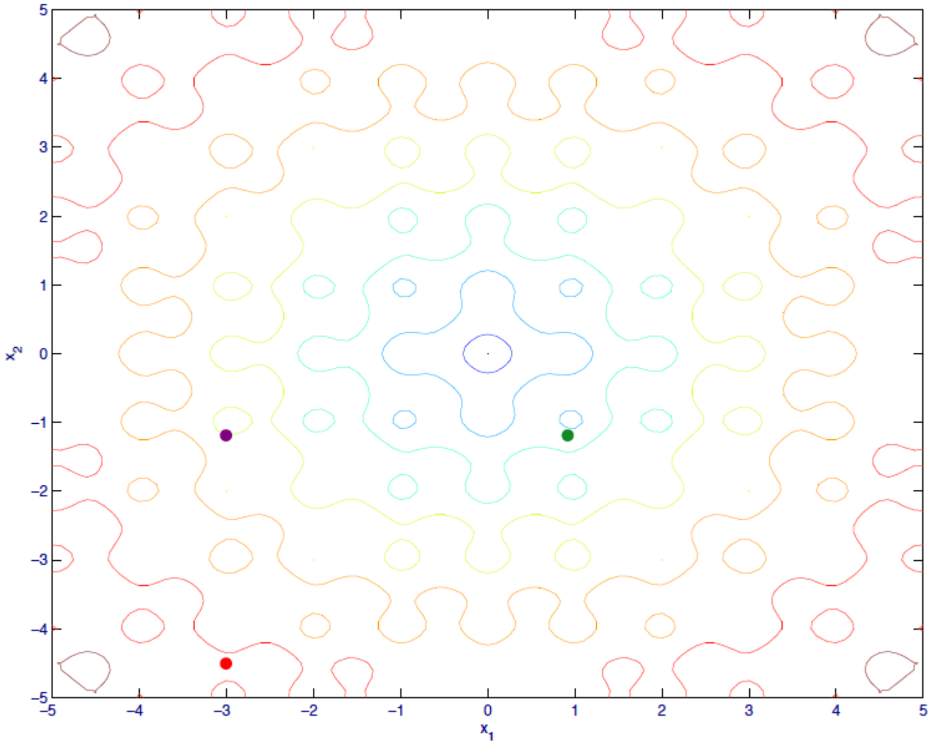
例子

构建捐赠向量



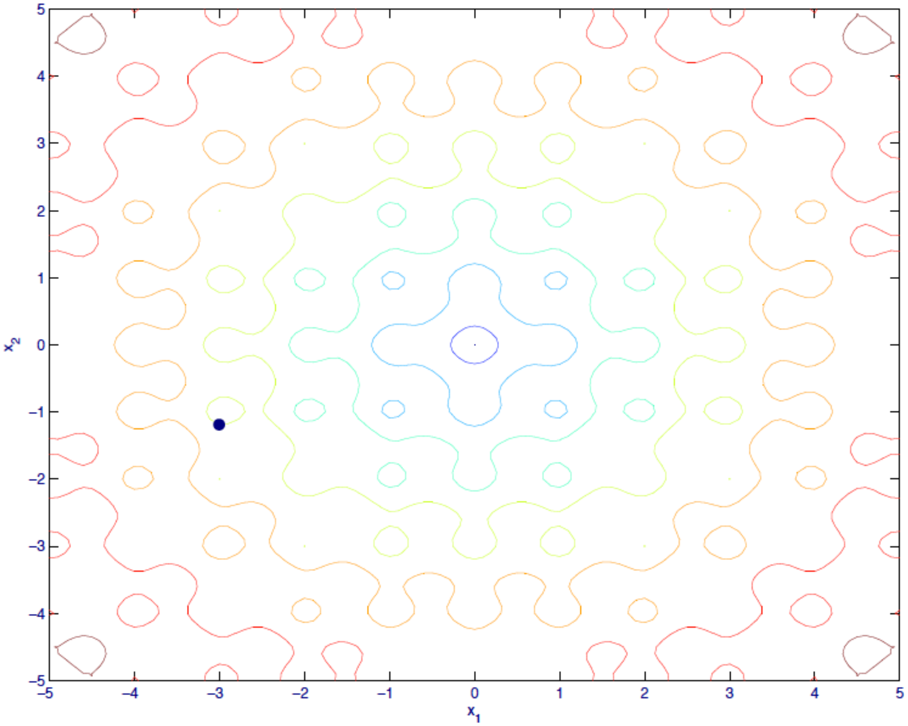
例子

由靶向量、捐赠向量生成试验向量



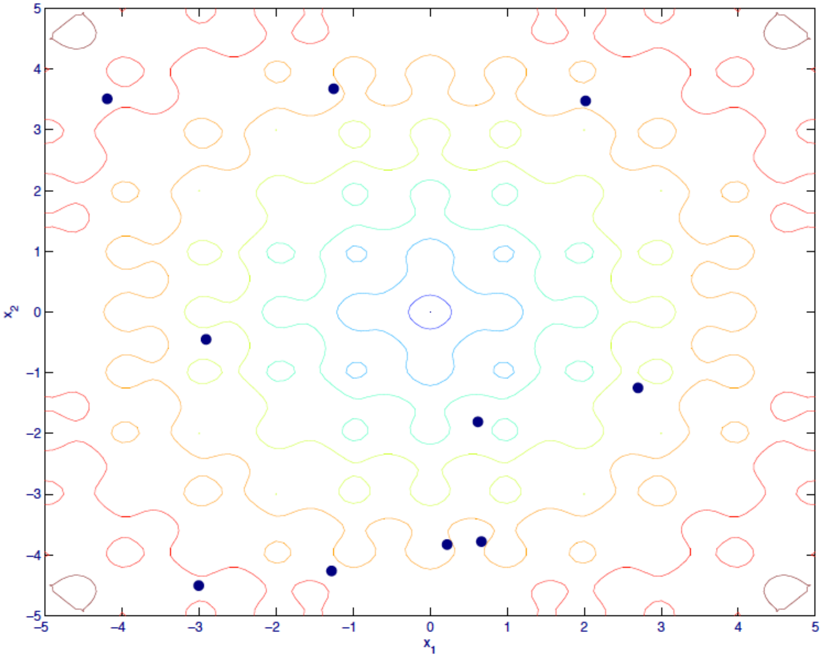
例子

在靶向量、试验向量中选择一个进入下一代

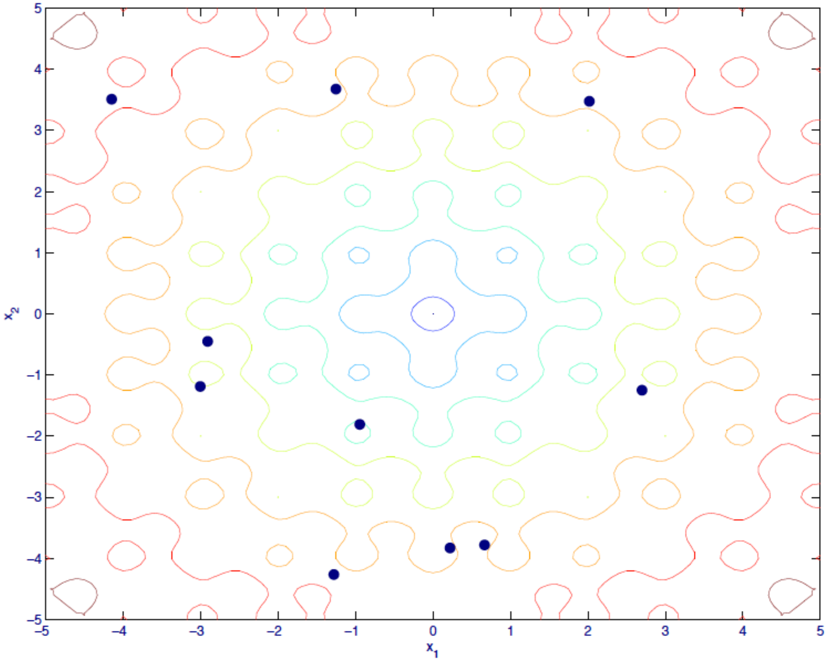


例子

第一代


5
4
3
2
1
0
−1
−2
−3
−4
−5
x_2
−5
−4
−3
−2
−1
0
1
2
3
4
5
x_1

第二代


5
4
3
2
1
0
−1
−2
−3
−4
−5
x_2
−5
−4
−3
−2
−1
0
1
2
3
4
5
x_1

群体智能之蚁群优化算法

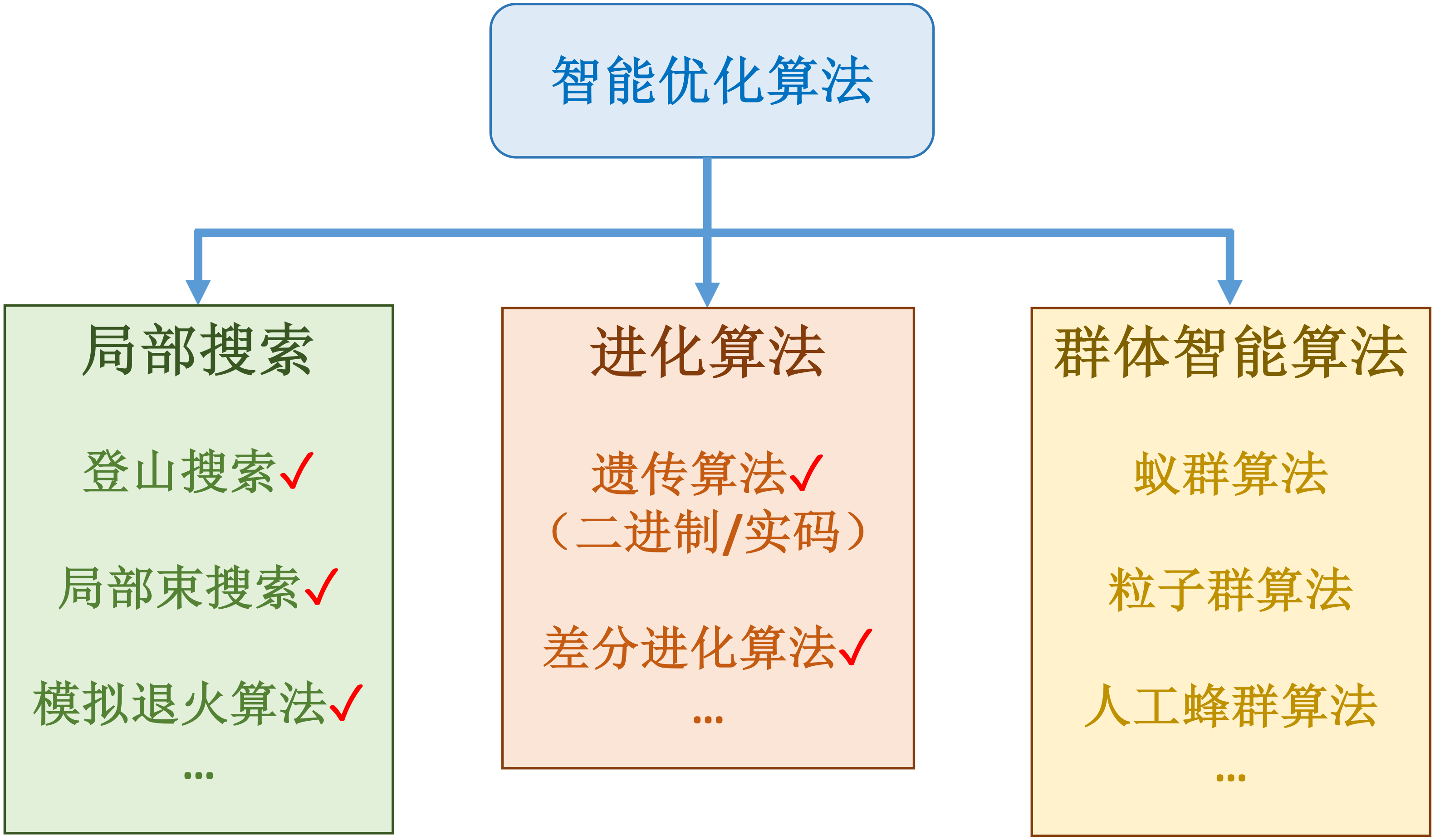
Part1 如何从复杂度的角度衡量算法好坏？

Part2 针对一些有特定形式的最优化问题，如何找到最优解？

Part3 针对较复杂的最优化问题（如NP难问题），如何找到较好解？

局部搜索、模拟退火、遗传算法、差分进化算法、蚁群算法、粒子群优化算法、人工蜂群算法、贝叶斯优化

Part4 针对较复杂的多目标最优化问题，如何找到较好解？



智能优化算法

局部搜索

登山搜索√

局部束搜索√

模拟退火算法√

…

进化算法

遗传算法√

（二进制/实码）

差分进化算法√

…

群体智能算法

蚁群算法

粒子群算法

人工蜂群算法

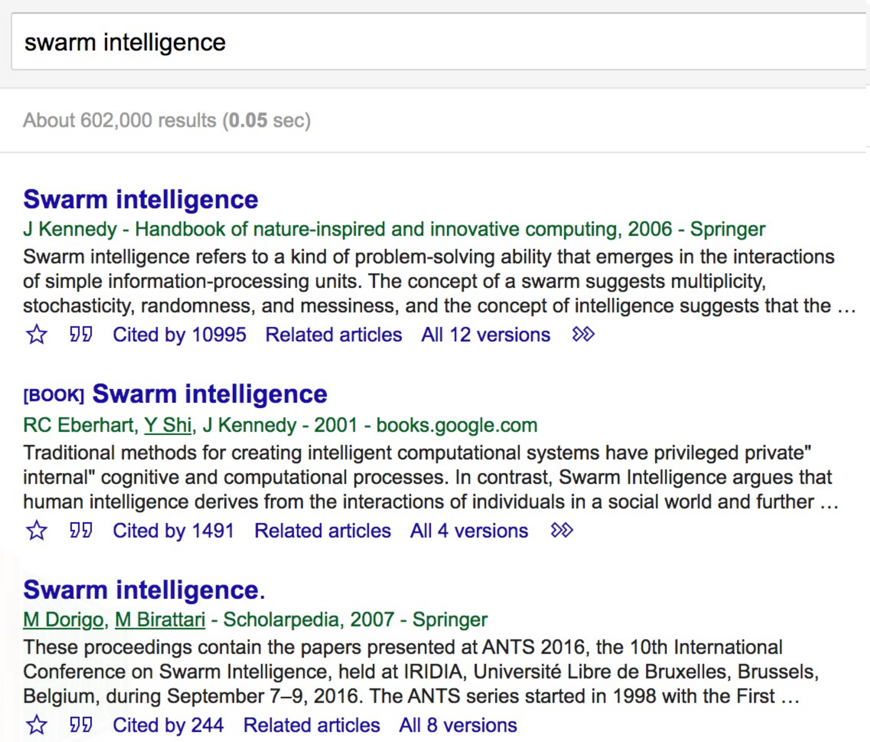
…

群体智能算法

“群体智能”的两种不同含义

—— swarm intelligence（swarm: 指动物/昆虫的群体）

—— crowd intelligence（crowd: 指人群）



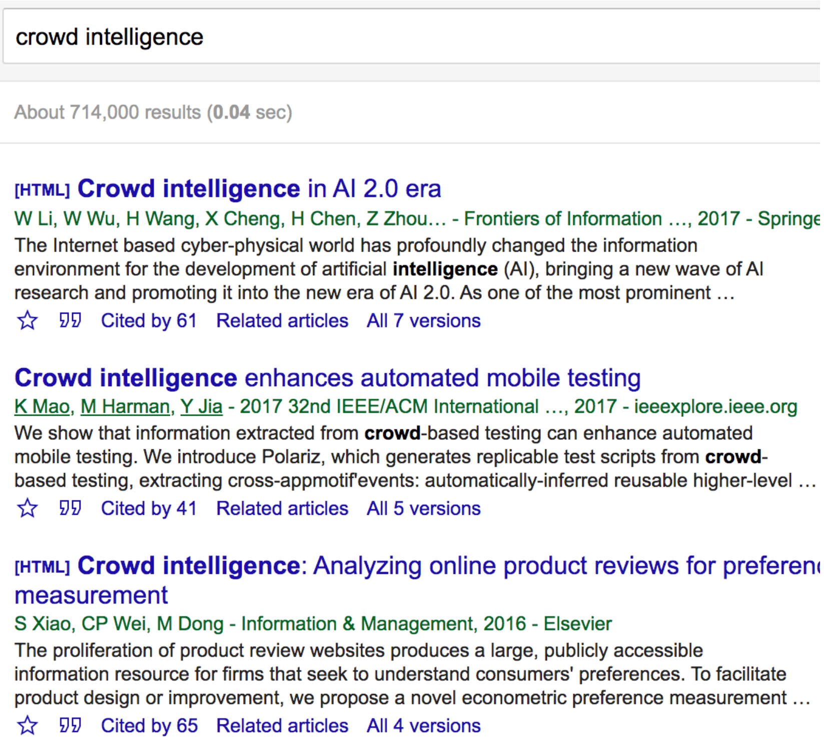
swarm intelligence

About 602,000 results (**0.05** sec)

Swarm intelligence
J Kennedy - Handbook of nature-inspired and innovative computing, 2006 - Springer
Swarm intelligence refers to a kind of problem-solving ability that emerges in the interactions of simple information-processing units. The concept of a swarm suggests multiplicity, stochasticity, randomness, and messiness, and the concept of intelligence suggests that the …
☆ ❞ Cited by 10995 Related articles All 12 versions »

[BOOK] **Swarm intelligence**
RC Eberhart, Y Shi, J Kennedy - 2001 - books.google.com
Traditional methods for creating intelligent computational systems have privileged private" internal" cognitive and computational processes. In contrast, Swarm Intelligence argues that human intelligence derives from the interactions of individuals in a social world and further …
☆ ❞ Cited by 1491 Related articles All 4 versions »

Swarm intelligence.
M Dorigo, M Birattari - Scholarpedia, 2007 - Springer
These proceedings contain the papers presented at ANTS 2016, the 10th International Conference on Swarm Intelligence, held at IRIDIA, Université Libre de Bruxelles, Brussels, Belgium, during September 7–9, 2016. The ANTS series started in 1998 with the First …
☆ ❞ Cited by 244 Related articles All 8 versions



crowd intelligence

About 714,000 results (**0.04** sec)

[HTML] **Crowd intelligence** in AI 2.0 era
W Li, W Wu, H Wang, X Cheng, H Chen, Z Zhou… - Frontiers of Information …, 2017 - Springe
The Internet based cyber-physical world has profoundly changed the information environment for the development of artificial **intelligence** (AI), bringing a new wave of AI research and promoting it into the new era of AI 2.0. As one of the most prominent …
☆ ❞ Cited by 61 Related articles All 7 versions

Crowd intelligence enhances automated mobile testing
K Mao, M Harman, Y Jia - 2017 32nd IEEE/ACM International …, 2017 - ieeexplore.ieee.org
We show that information extracted from **crowd**-based testing can enhance automated mobile testing. We introduce Polariz, which generates replicable test scripts from **crowd**-based testing, extracting cross-appmotif'events: automatically-inferred reusable higher-level …
☆ ❞ Cited by 41 Related articles All 5 versions

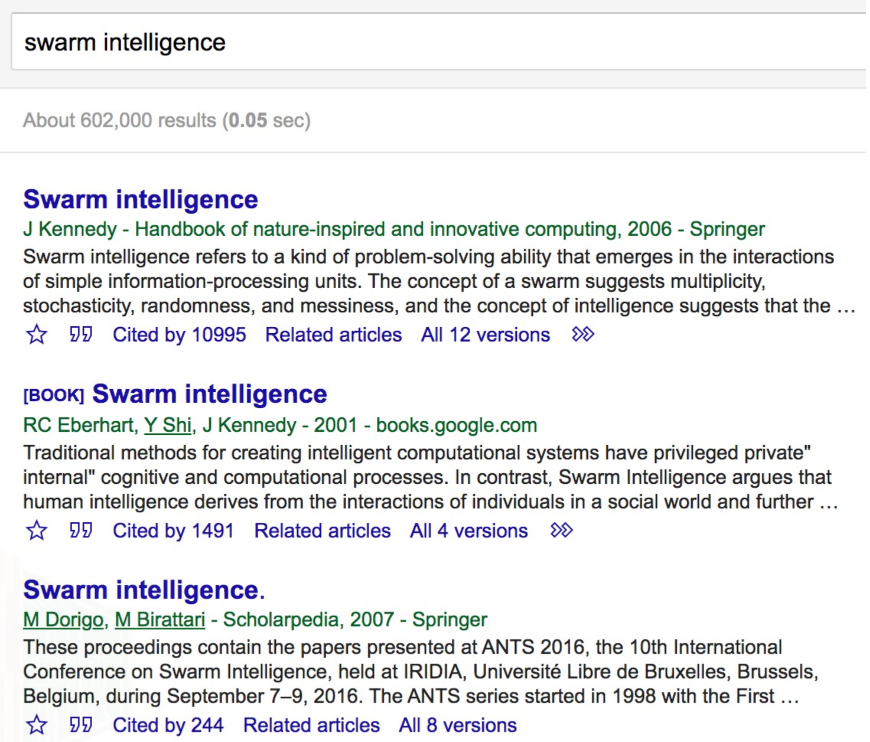
[HTML] **Crowd intelligence**: Analyzing online product reviews for preferen
measurement
S Xiao, CP Wei, M Dong - Information & Management, 2016 - Elsevier
The proliferation of product review websites produces a large, publicly accessible information resource for firms that seek to understand consumers' preferences. To facilitate product design or improvement, we propose a novel econometric preference measurement …
☆ ❞ Cited by 65 Related articles All 4 versions

群体智能算法

“群体智能”的两种不同含义

—— swarm intelligence（swarm: 指动物/昆虫的群体）

通过模拟昆虫、兽群、鸟群、鱼群的群集行为搜索解（智能优化算法）



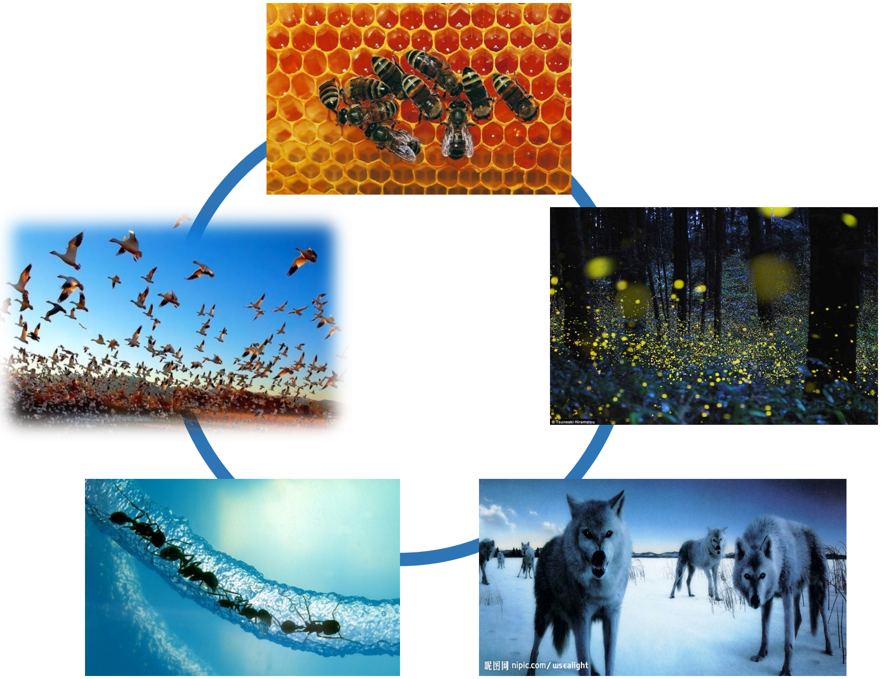
swarm intelligence

About 602,000 results (**0.05** sec)

Swarm intelligence
J Kennedy - Handbook of nature-inspired and innovative computing, 2006 - Springer
Swarm intelligence refers to a kind of problem-solving ability that emerges in the interactions of simple information-processing units. The concept of a swarm suggests multiplicity, stochasticity, randomness, and messiness, and the concept of intelligence suggests that the …
☆ ❞ Cited by 10995 Related articles All 12 versions »

[BOOK] **Swarm intelligence**
RC Eberhart, Y Shi, J Kennedy - 2001 - books.google.com
Traditional methods for creating intelligent computational systems have privileged private" internal" cognitive and computational processes. In contrast, Swarm Intelligence argues that human intelligence derives from the interactions of individuals in a social world and further …
☆ ❞ Cited by 1491 Related articles All 4 versions »

Swarm intelligence.
M Dorigo, M Birattari - Scholarpedia, 2007 - Springer
These proceedings contain the papers presented at ANTS 2016, the 10th International Conference on Swarm Intelligence, held at IRIDIA, Université Libre de Bruxelles, Brussels, Belgium, during September 7–9, 2016. The ANTS series started in 1998 with the First …
☆ ❞ Cited by 244 Related articles All 8 versions

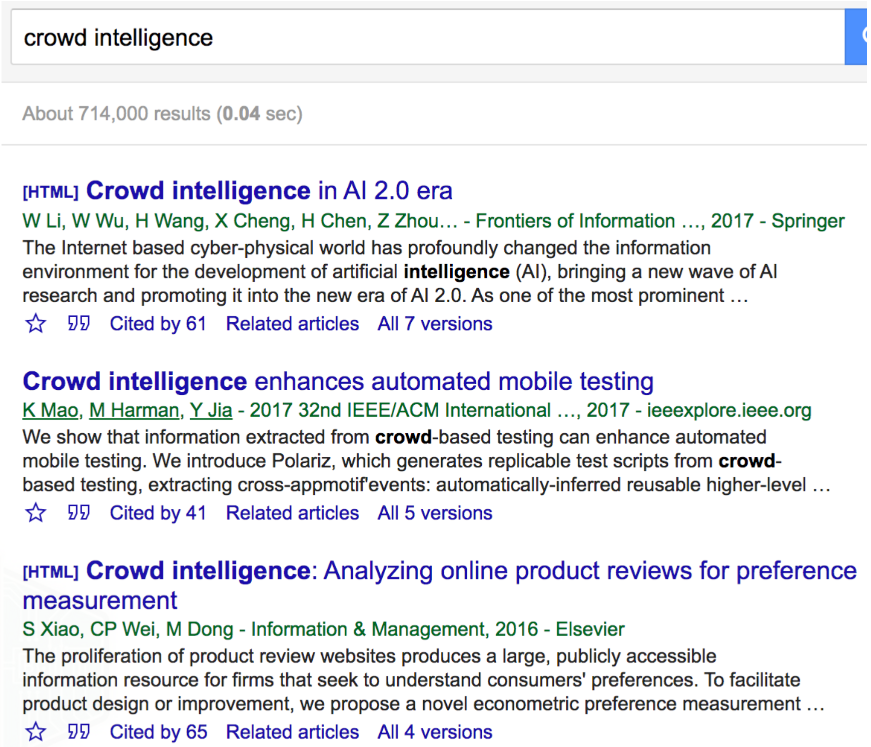


群体智能算法

“群体智能”的两种不同含义

—— crowd intelligence（crowd: 指人群）

利用众人参与的形式完成特定任务（众包）



crowd intelligence

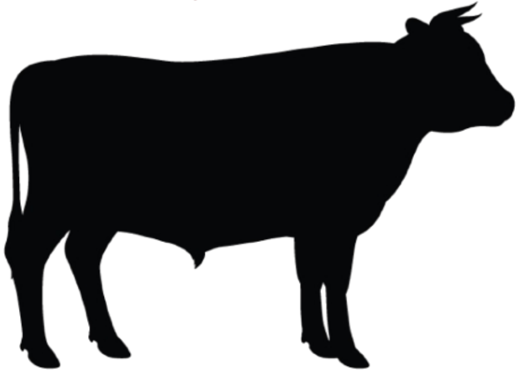
About 714,000 results (**0.04** sec)

[HTML] **Crowd intelligence** in AI 2.0 era
W Li, W Wu, H Wang, X Cheng, H Chen, Z Zhou… - Frontiers of Information …, 2017 - Springer
The Internet based cyber-physical world has profoundly changed the information environment for the development of artificial **intelligence** (AI), bringing a new wave of AI research and promoting it into the new era of AI 2.0. As one of the most prominent …
☆ 99 Cited by 61 Related articles All 7 versions

Crowd intelligence enhances automated mobile testing
K Mao, M Harman, Y Jia - 2017 32nd IEEE/ACM International …, 2017 - ieeexplore.ieee.org
We show that information extracted from **crowd**-based testing can enhance automated mobile testing. We introduce Polariz, which generates replicable test scripts from **crowd**-based testing, extracting cross-appmotif'events: automatically-inferred reusable higher-level …
☆ 99 Cited by 41 Related articles All 5 versions

[HTML] **Crowd intelligence**: Analyzing online product reviews for preference measurement
S Xiao, CP Wei, M Dong - Information & Management, 2016 - Elsevier
The proliferation of product review websites produces a large, publicly accessible information resource for firms that seek to understand consumers' preferences. To facilitate product design or improvement, we propose a novel econometric preference measurement …
☆ 99 Cited by 65 Related articles All 4 versions

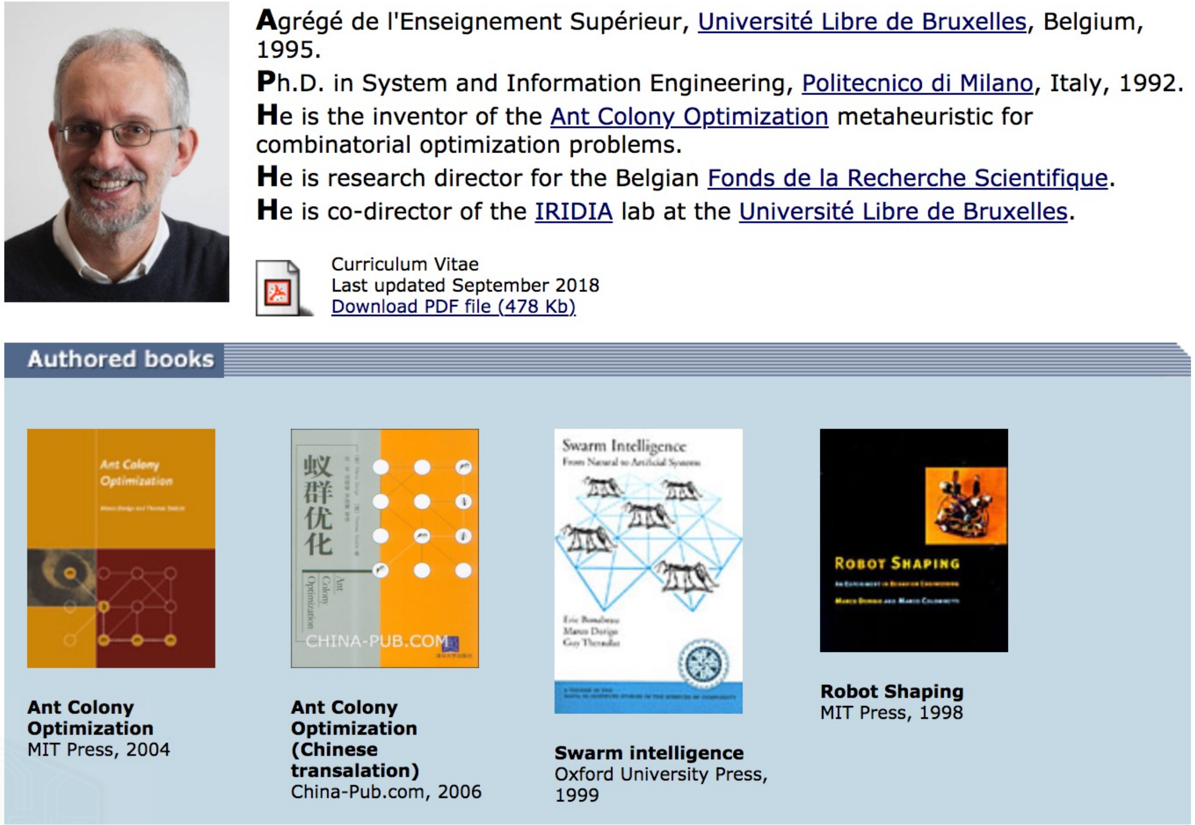
Actual weight of the ox on show:
1,198 lbs



Average guess of 787 people who took part:
1,207 lbs

蚁群优化算法（Ant colony optimization algorithm）

1992年Marco Dorigo在其博士学位论文中首次介绍



Agrégé de l'Enseignement Supérieur, Université Libre de Bruxelles, Belgium, 1995.

Ph.D. in System and Information Engineering, Politecnico di Milano, Italy, 1992.

He is the inventor of the Ant Colony Optimization metaheuristic for combinatorial optimization problems.

He is research director for the Belgian Fonds de la Recherche Scientifique.

He is co-director of the IRIDIA lab at the Université Libre de Bruxelles.

Curriculum Vitae
Last updated September 2018
Download PDF file (478 Kb)

Authored books

Ant Colony Optimization
MIT Press, 2004

Ant Colony Optimization (Chinese transalation)
China-Pub.com, 2006

Swarm intelligence
Oxford University Press, 1999

Robot Shaping
MIT Press, 1998

Marco Dorigo
的Google Scholar
总引约13万次

蚁群优化算法

蚂蚁寻食物

蚂蚁模拟 Colony simulation

https://www.bilibili.com/video/BV1JB4y1P7mk?from=search&seid=5718585914166882441

蚂蚁寻食机制

通过分别与环境交互的方式实现间接通信

蚂蚁视力不好，在爬过的地方留下信息素（pheromone）

通过信息素实现间接通信

某路径信息素浓度越高，后续蚂蚁选择该路径概率越大

双桥实验

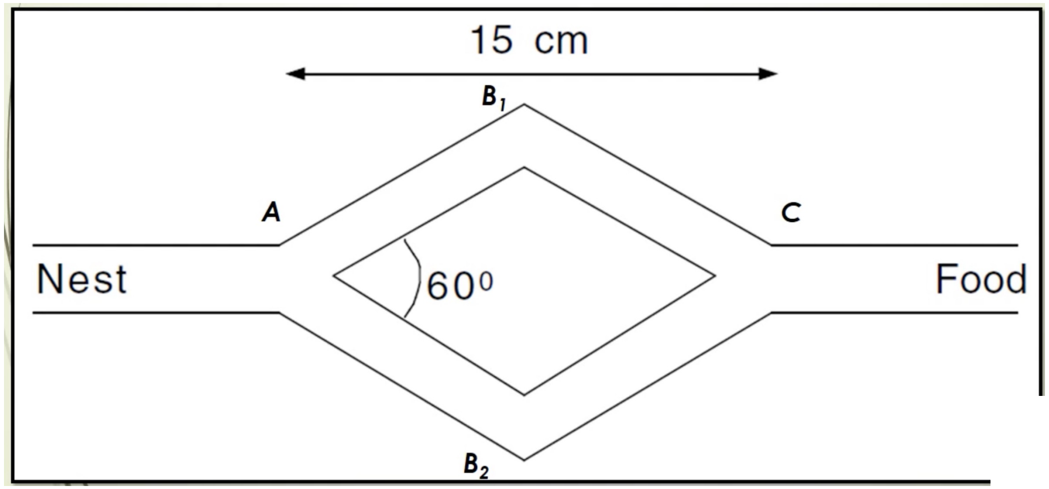
分析信息素对蚂蚁路径选择的影响

实验假设：

（1）所有蚂蚁爬行速度一样；（2）所有蚂蚁释放等量的信息素；

（3）蚂蚁在往返同一路径时释放的信息素没有差别

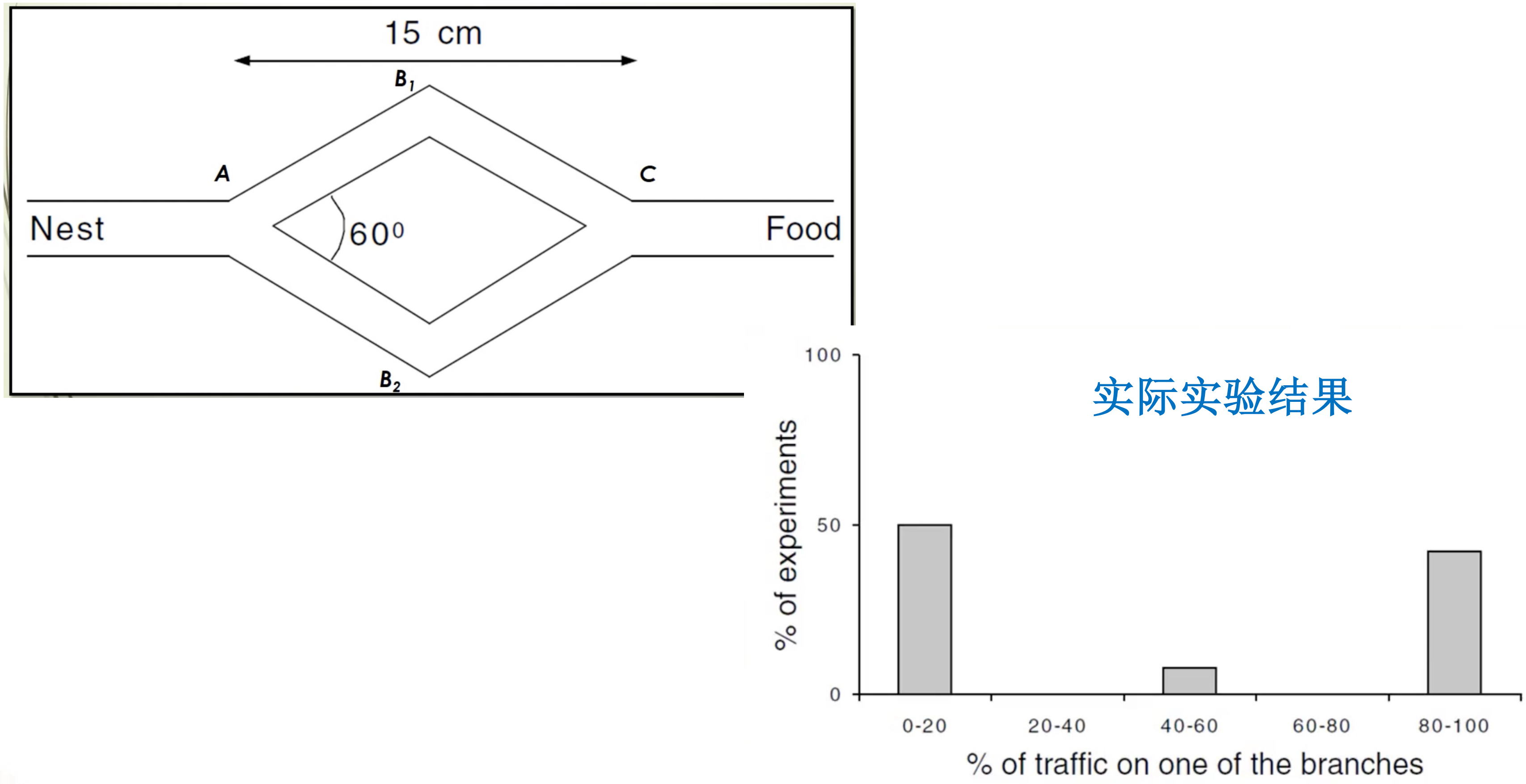
双桥实验



设置1

内容取自Jeff Lettman “An Introduction to Ant colony optimization”

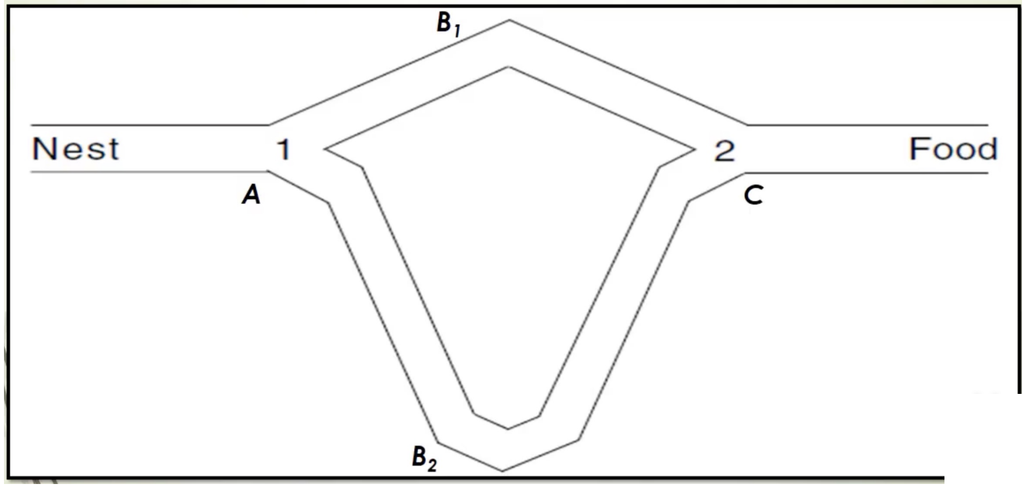
双桥实验



实际实验结果

内容取自Jeff Lettman “An Introduction to Ant colony optimization”

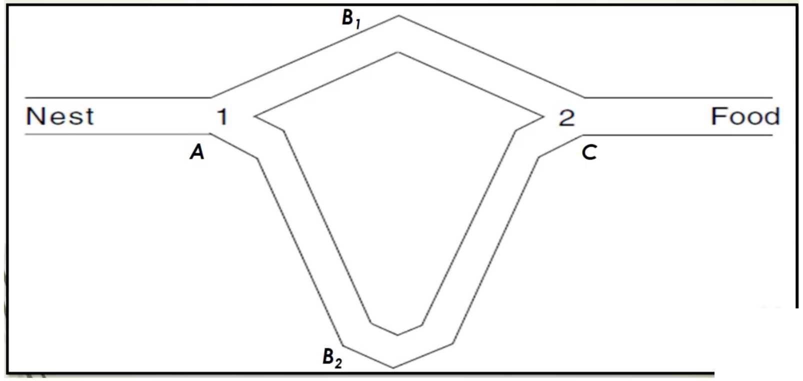
双桥实验

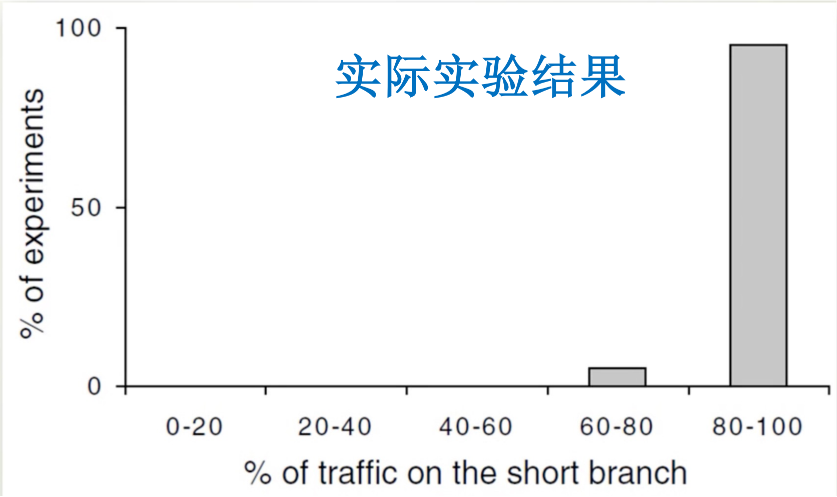


设置2

内容取自Jeff Lettman “An Introduction to Ant colony optimization”

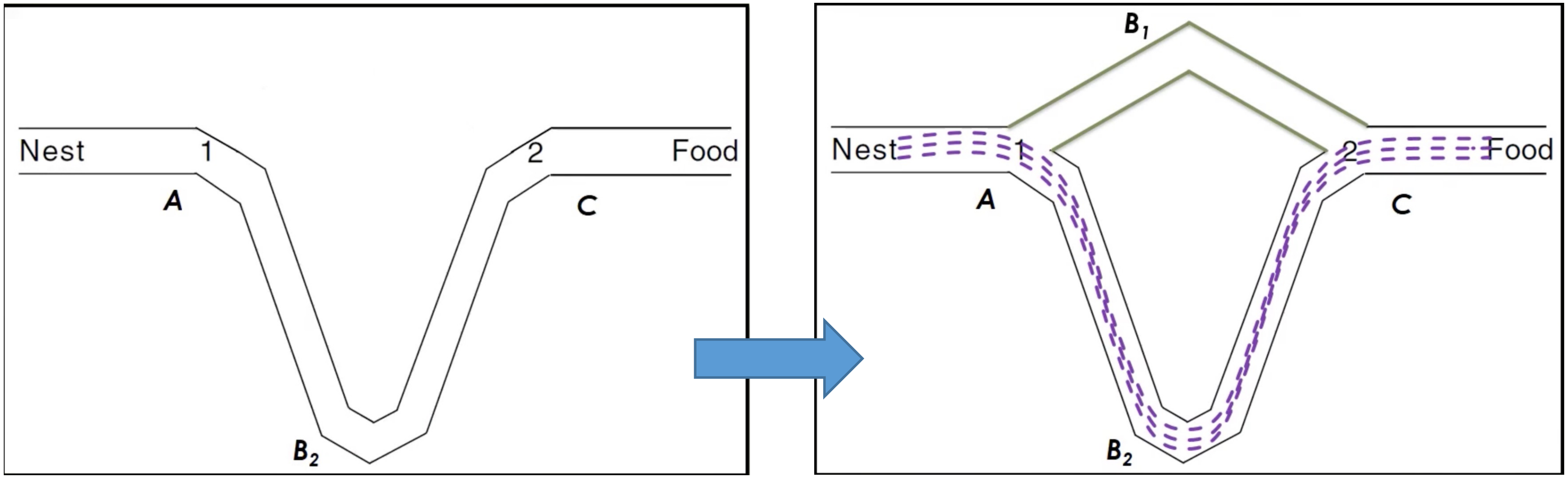
双桥实验





内容取自Jeff Lettman “An Introduction to Ant colony optimization”

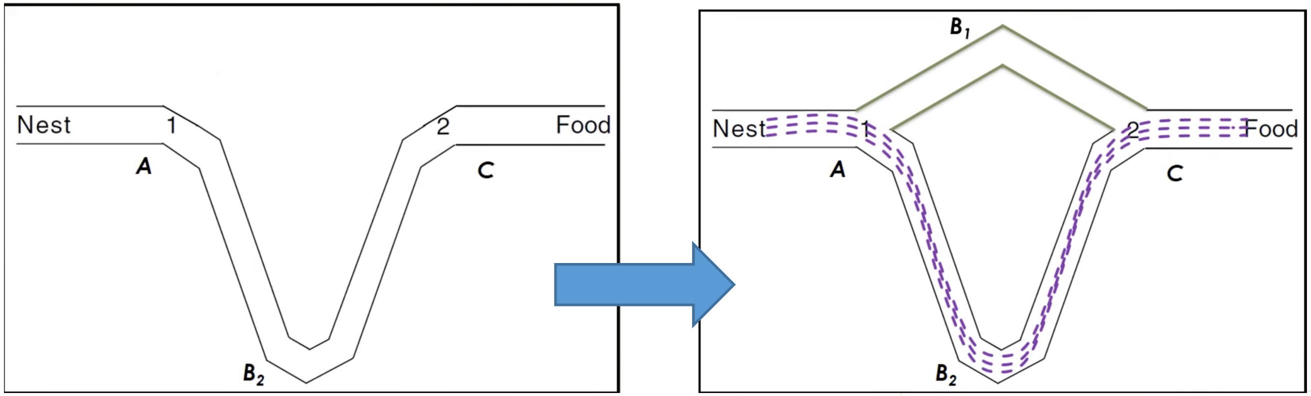
双桥实验

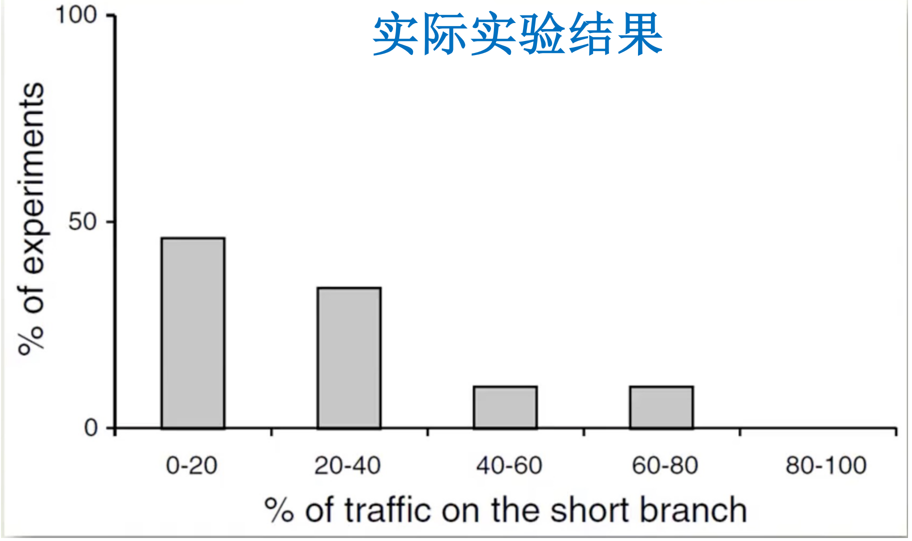


设置3

内容取自Jeff Lettman “An Introduction to Ant colony optimization”

双桥实验

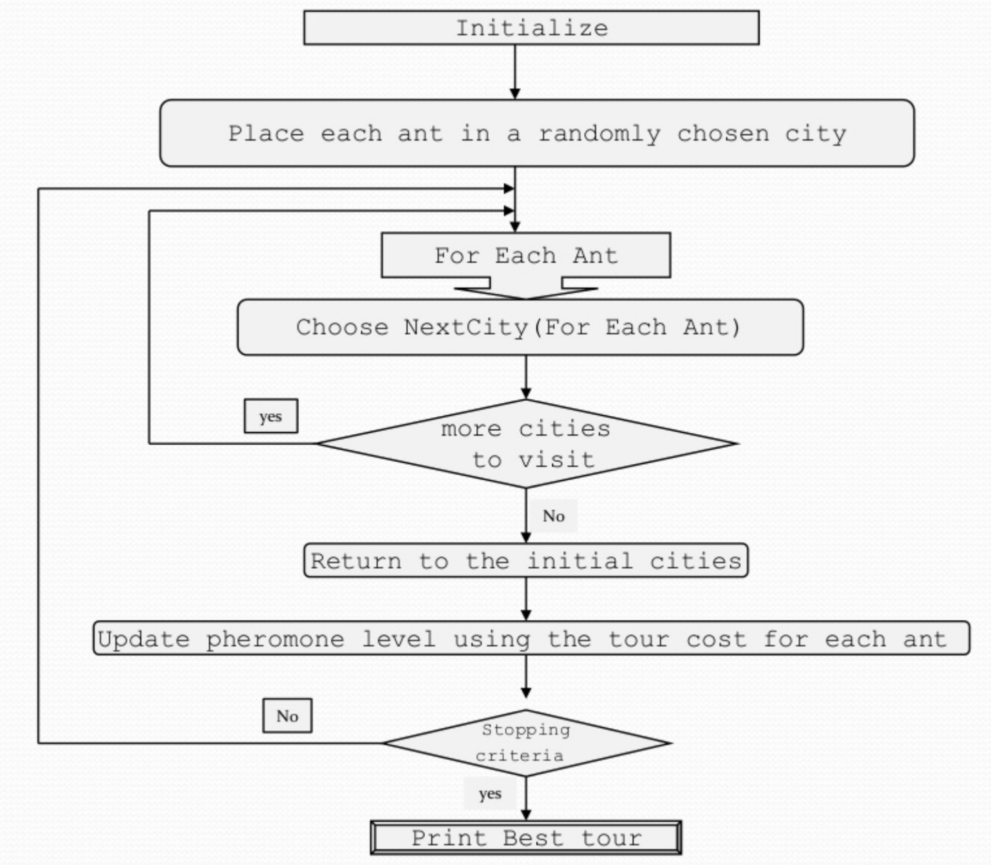



实际实验结果

内容取自Jeff Lettman “An Introduction to Ant colony optimization”

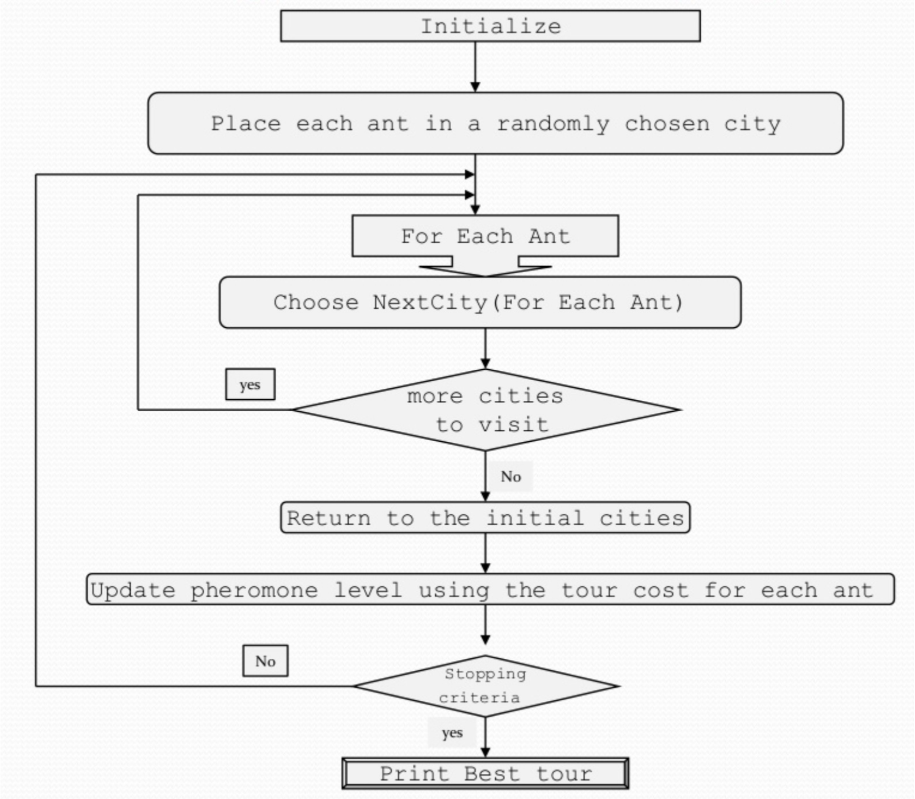
蚁群系统

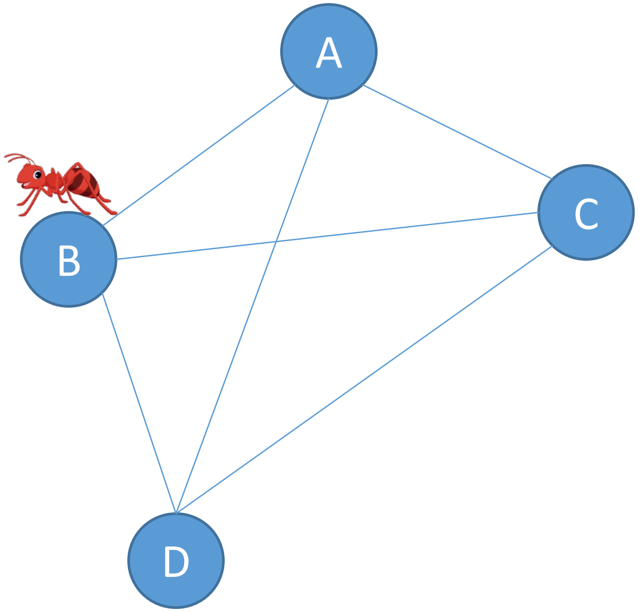
蚁群系统（ant system）是第一个蚁群优化算法，被用于解决旅行商问题


Initialize
Place each ant in a randomly chosen city
For Each Ant
Choose NextCity(For Each Ant)
yes
more cities to visit
No
Return to the initial cities
Update pheromone level using the tour cost for each ant
No
Stopping criteria
yes
Print Best tour

图片取自Joy Dutta课件“Ant colony optimization”

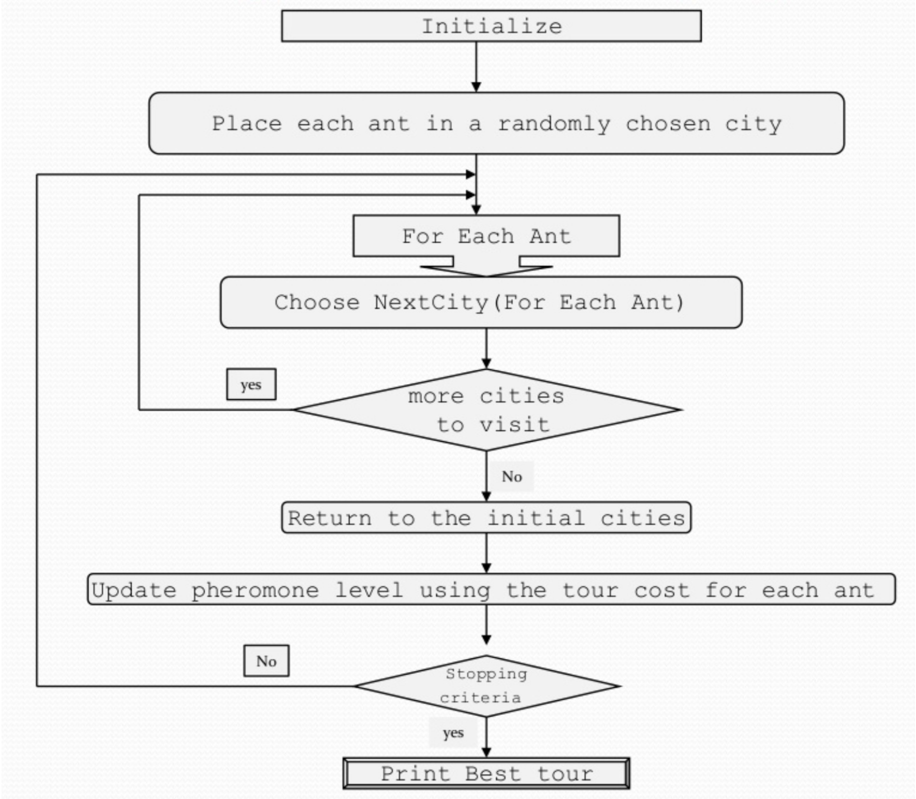
蚁群系统


Initialize
Place each ant in a randomly chosen city
For Each Ant
Choose NextCity(For Each Ant)
yes
more cities to visit
No
Return to the initial cities
Update pheromone level using the tour cost for each ant
No
Stopping criteria
yes
Print Best tour



图片取自Joy Dutta课件“Ant colony optimization”

蚁群系统



Initialize

Place each ant in a randomly chosen city

For Each Ant

Choose NextCity(For Each Ant)

more cities to visit

yes

No

Return to the initial cities

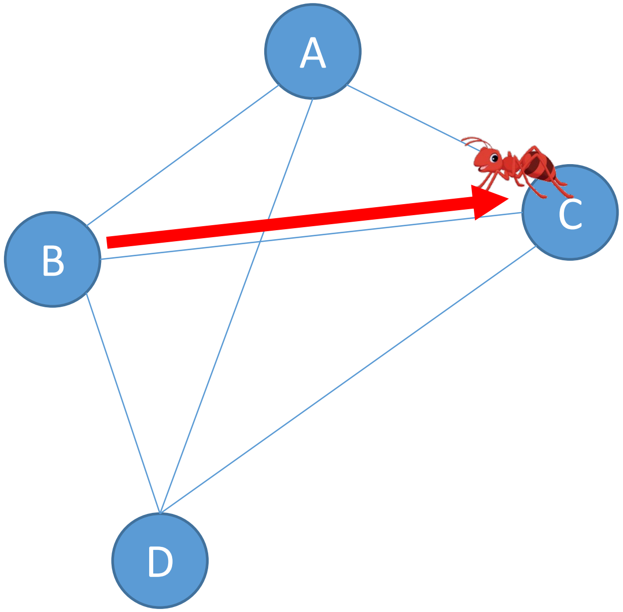
Update pheromone level using the tour cost for each ant

Stopping criteria

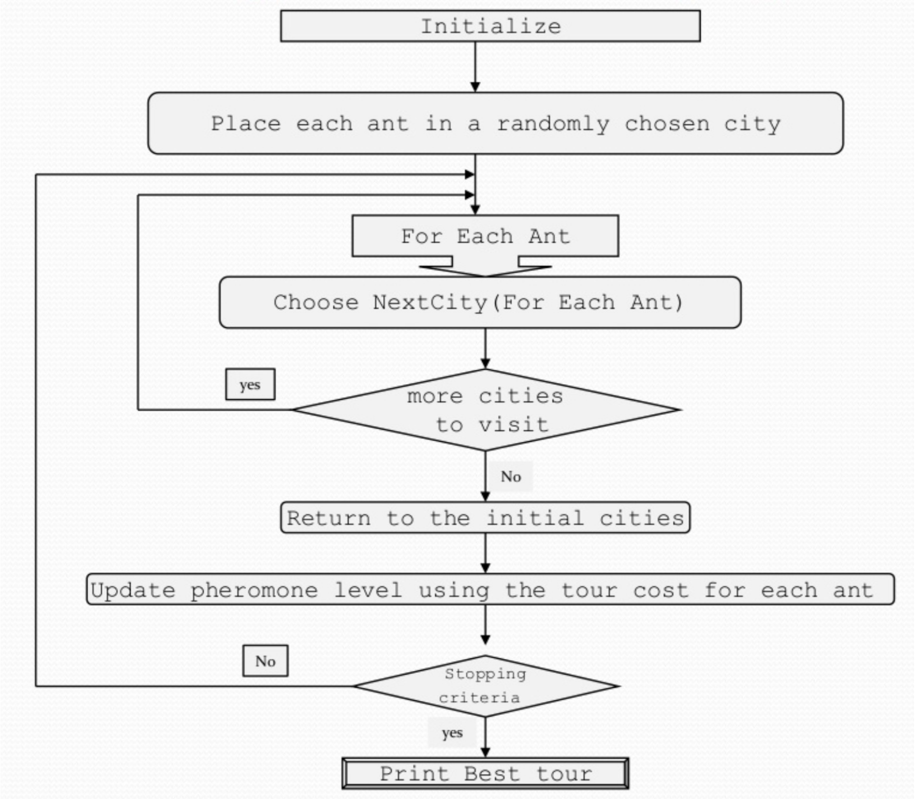
No

yes

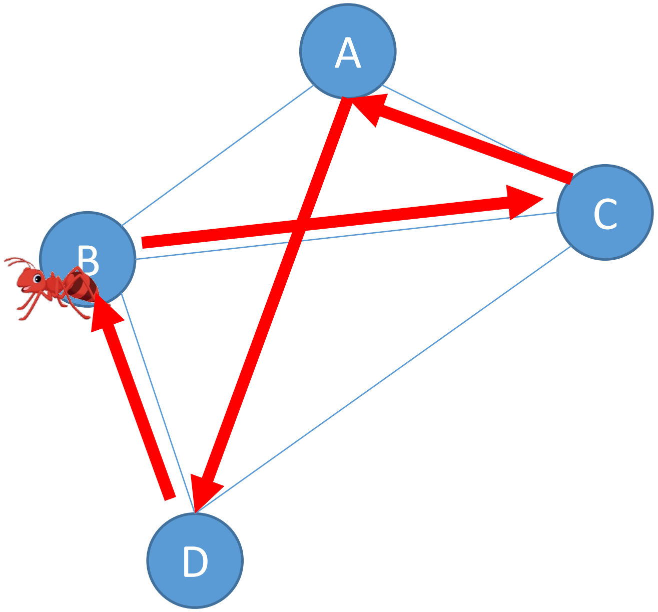
Print Best tour



蚁群系统

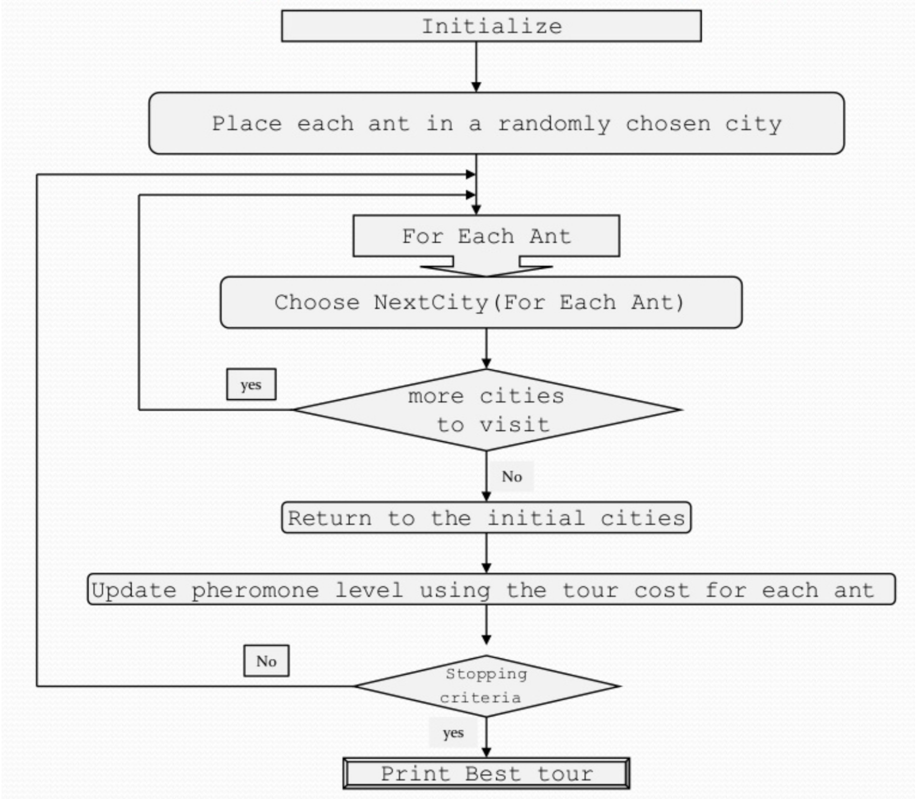


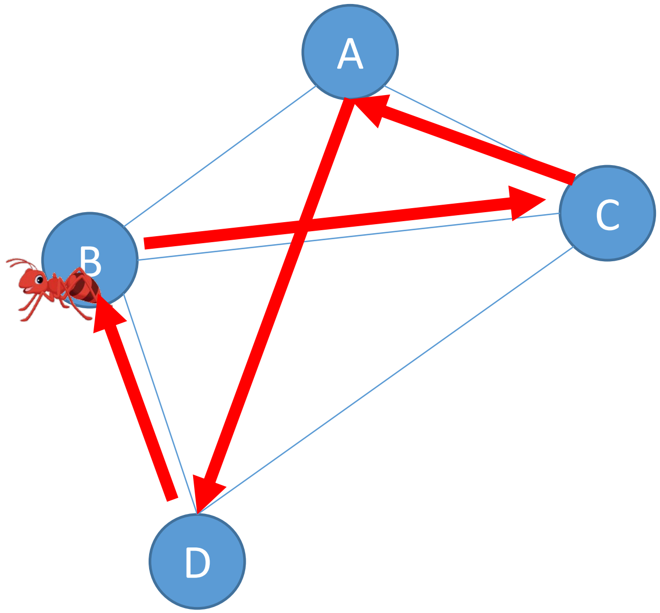
Initialize
Place each ant in a randomly chosen city
For Each Ant
Choose NextCity(For Each Ant)
yes
more cities to visit
No
Return to the initial cities
Update pheromone level using the tour cost for each ant
No
Stopping criteria
yes
Print Best tour



每只蚂蚁每次如何选择下一个城市？

蚁群系统


Initialize
Place each ant in a randomly chosen city
For Each Ant
Choose NextCity(For Each Ant)
yes
more cities to visit
No
Return to the initial cities
Update pheromone level using the tour cost for each ant
No
Stopping criteria
yes
Print Best tour



每只蚂蚁每次如何选择下一个城市？

$$p_k(r,s) = \frac{T(r,s)\cdot H(r,s)^{\beta}}{\sum_{\text{unvisited cities } c} T(r,c)\cdot H(r,c)^{\beta}}$$

k: 第k只蚂蚁

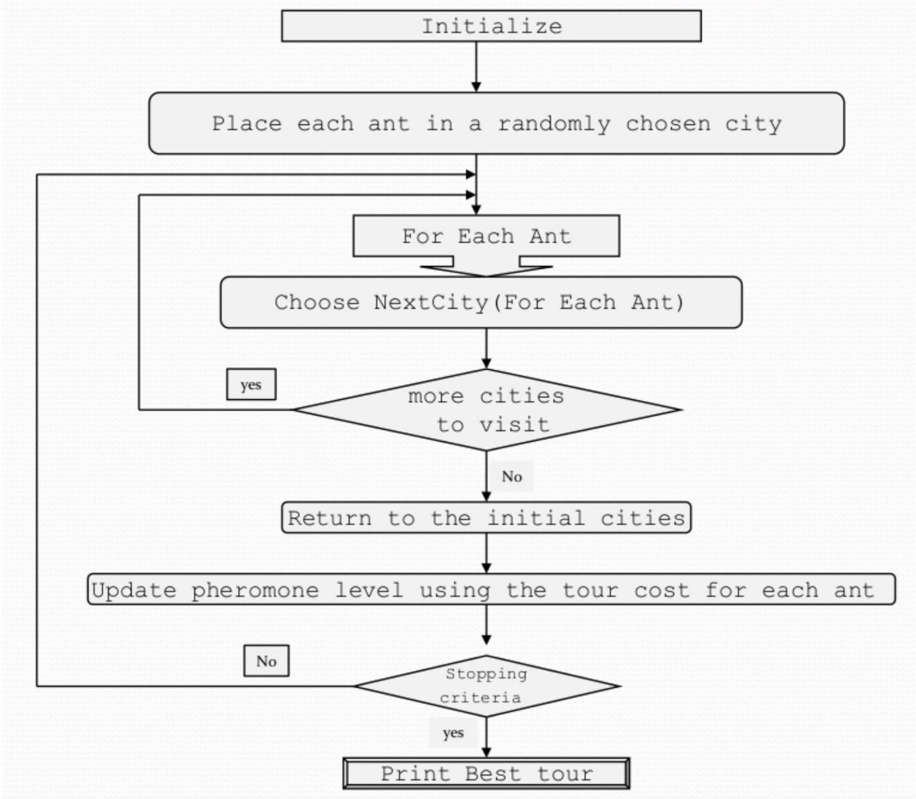
r: 现在城市； s和c: 可能选择的城市

T(r, s): 信息素浓度

H(r, s): 距离倒数; β: 控制参数

有的版本会对T(r, s)的指数位置加控制参数

蚁群系统



Initialize

Place each ant in a randomly chosen city

For Each Ant

Choose NextCity(For Each Ant)

more cities to visit

yes

No

Return to the initial cities

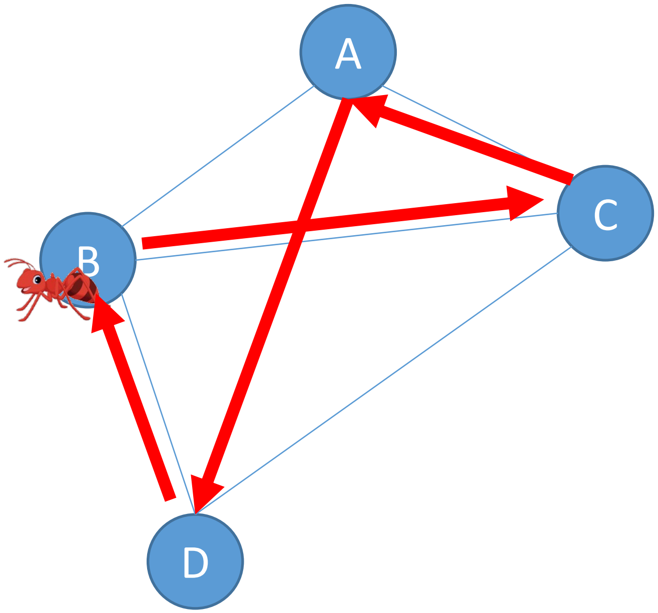
Update pheromone level using the tour cost for each ant

Stopping criteria

No

yes

Print Best tour



信息素浓度如何更新？

$$T\left(r,s\right) \leftarrow \rho \cdot T(r,s) + \sum_{k=1}^{m} A_k(r,s)$$

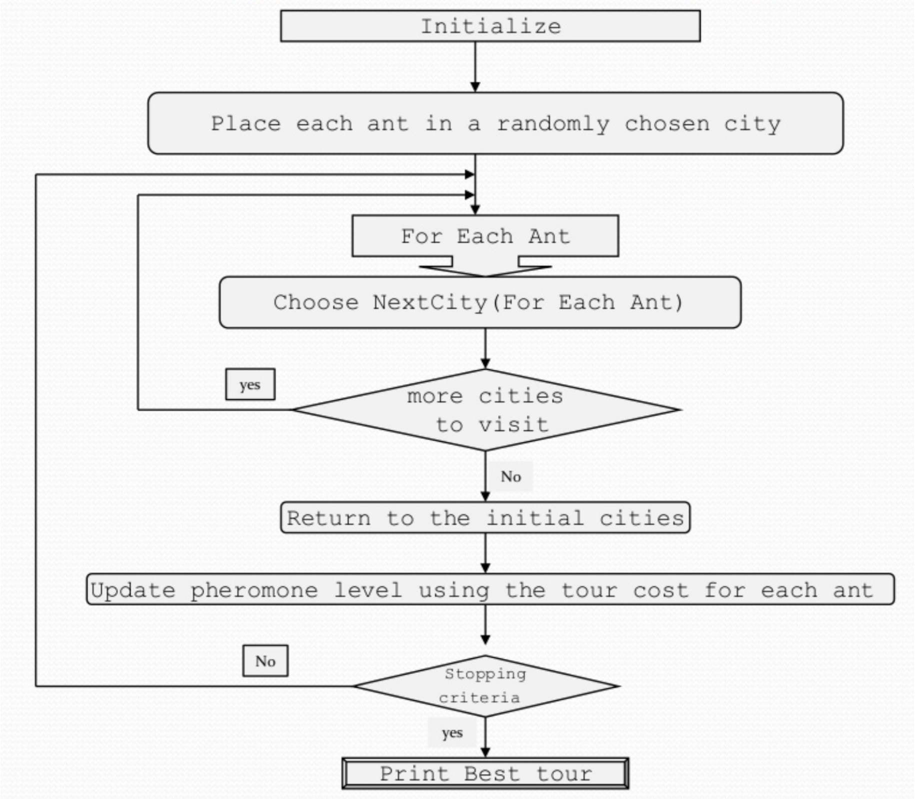
ρ: 衰减系数

m: 蚂蚁总数

$A_k(r, s)$: 如果蚂蚁k没走过rs，值为0

如果蚂蚁k走过rs，值为其总路程的倒数

蚁群系统


Initialize
Place each ant in a randomly chosen city
For Each Ant
Choose NextCity(For Each Ant)
yes
more cities to visit
No
Return to the initial cities
Update pheromone level using the tour cost for each ant
No
Stopping criteria
yes
Print Best tour

$$p_k(r,s) = \frac{T(r,s)\cdot H(r,s)^{\beta}}{\sum_{\text{unvisited cities } c} T(r,c)\cdot H(r,c)^{\beta}}$$

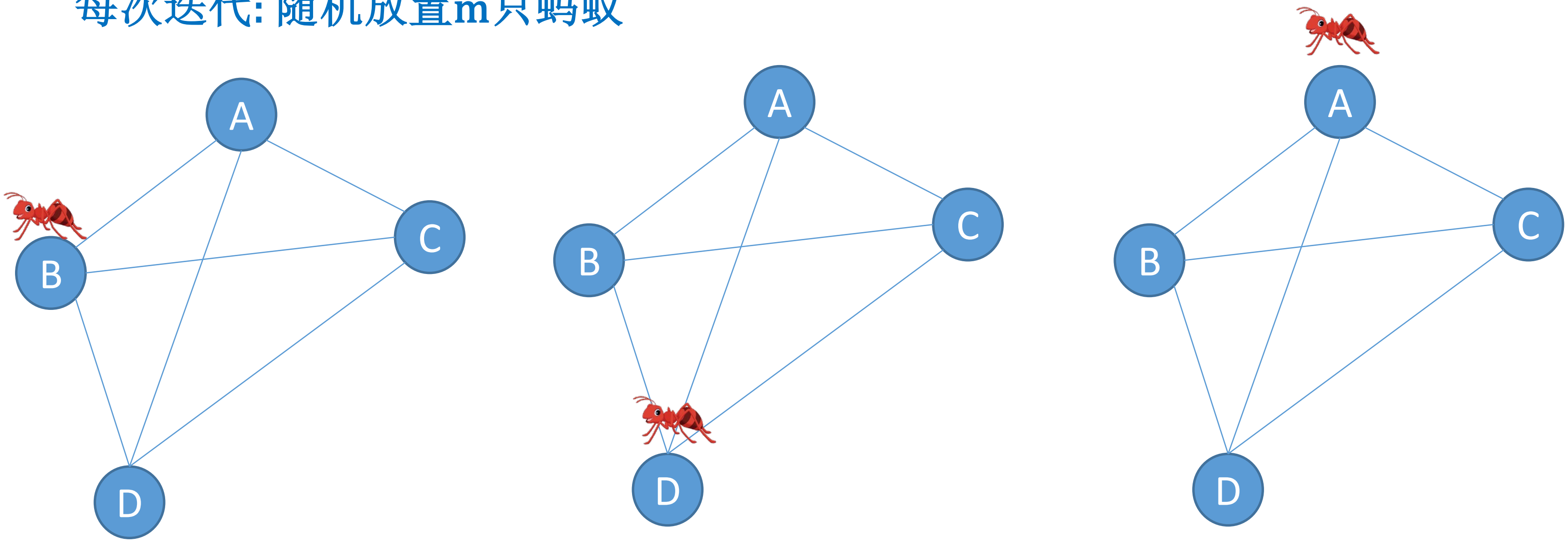
$$T\left(r,s\right) \leftarrow \rho \cdot T(r,s) + \sum_{k=1}^{m} A_k(r,s)$$

注意每次都是重新随机放置蚂蚁（相当于之前的蚂蚁死掉了）

与遗传算法、差分进化算法不同

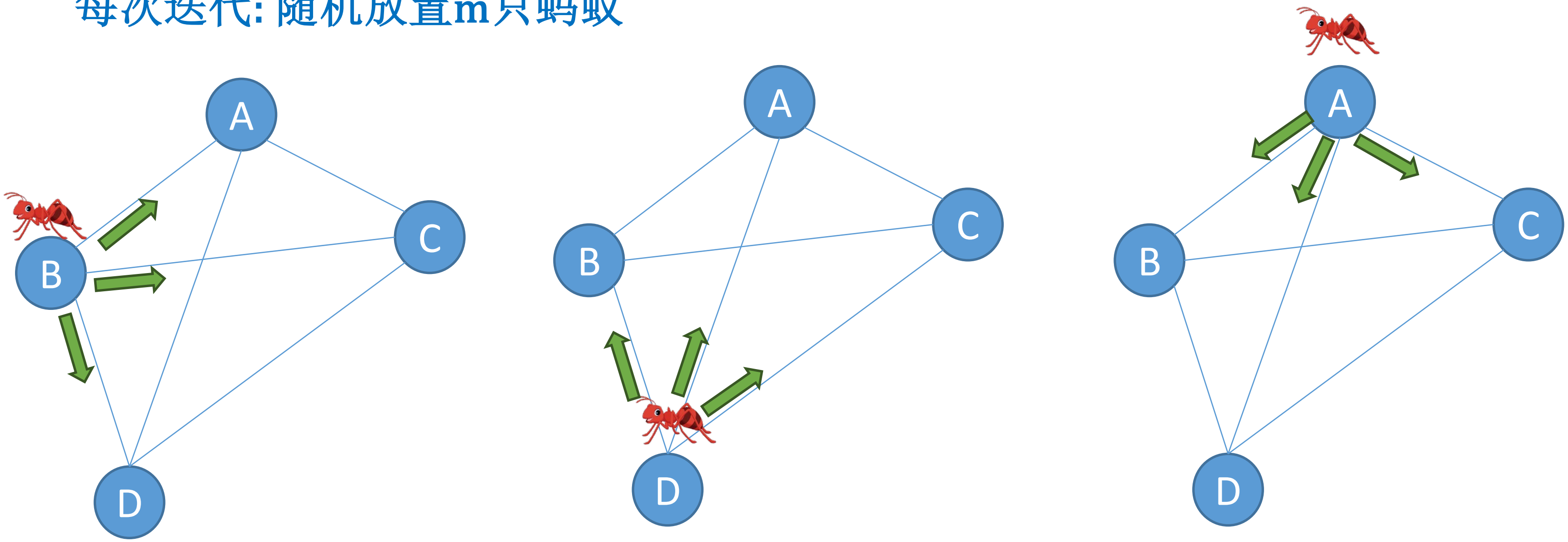
蚁群系统

每次迭代: 随机放置m只蚂蚁



蚁群系统

每次迭代: 随机放置m只蚂蚁



每只蚂蚁每次如何选择下一个城市？

$$p_k(r,s) = \frac{T(r,s) \cdot H(r,s)^{\beta}}{\sum_{\text{unvisited cities } c} T(r,c) \cdot H(r,c)^{\beta}}$$

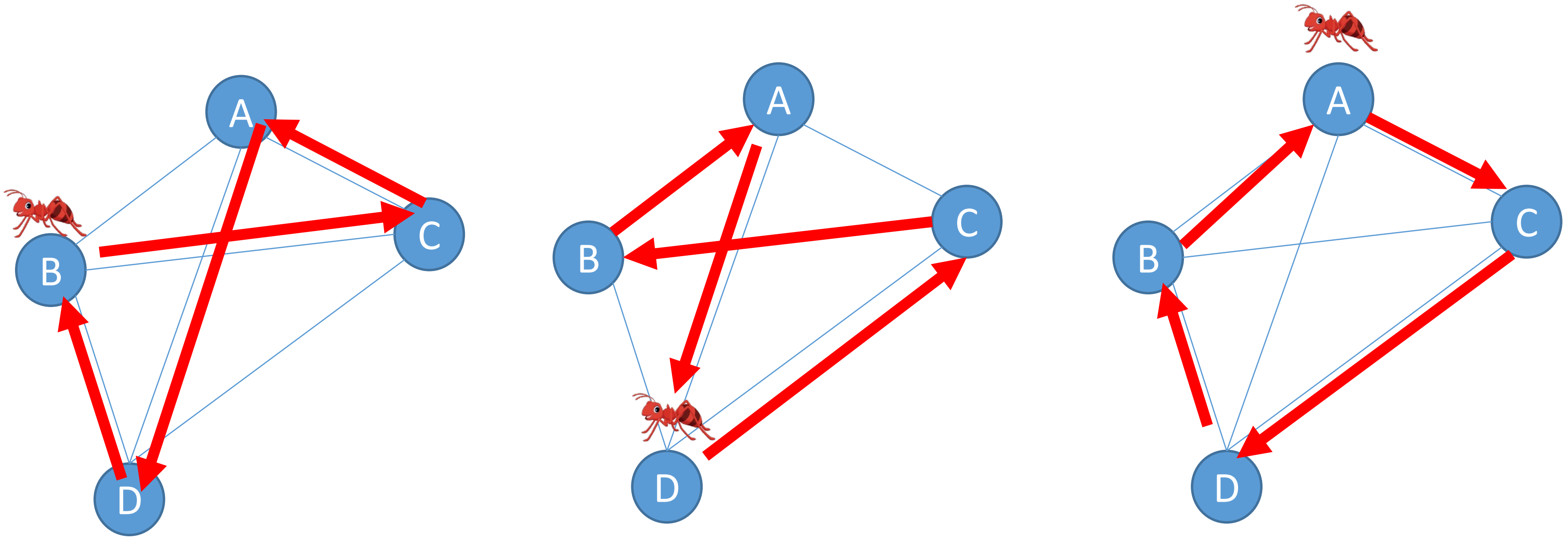
k: 第k只蚂蚁

r: 现在城市； s和c: 可能选择的城市

T(r, s): 信息素浓度

H(r, s): 距离倒数; β: 控制参数

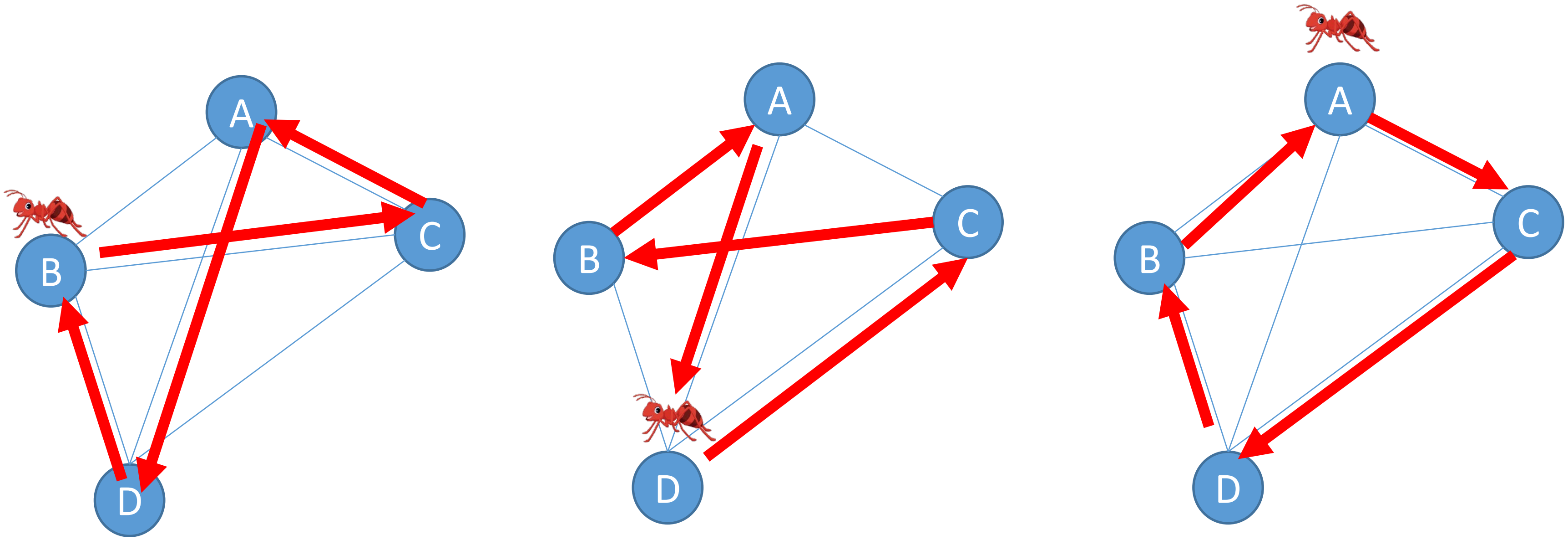
蚁群系统



根据各蚂蚁走过路径的总长更新相应各边的信息素浓度

思想: 路径总长越短，路径包含各边新增的信息素浓度越高

蚁群系统



$$T(r,s) \leftarrow \rho \cdot T(r,s) + \sum_{k=1}^{m} A_k(r,s)$$

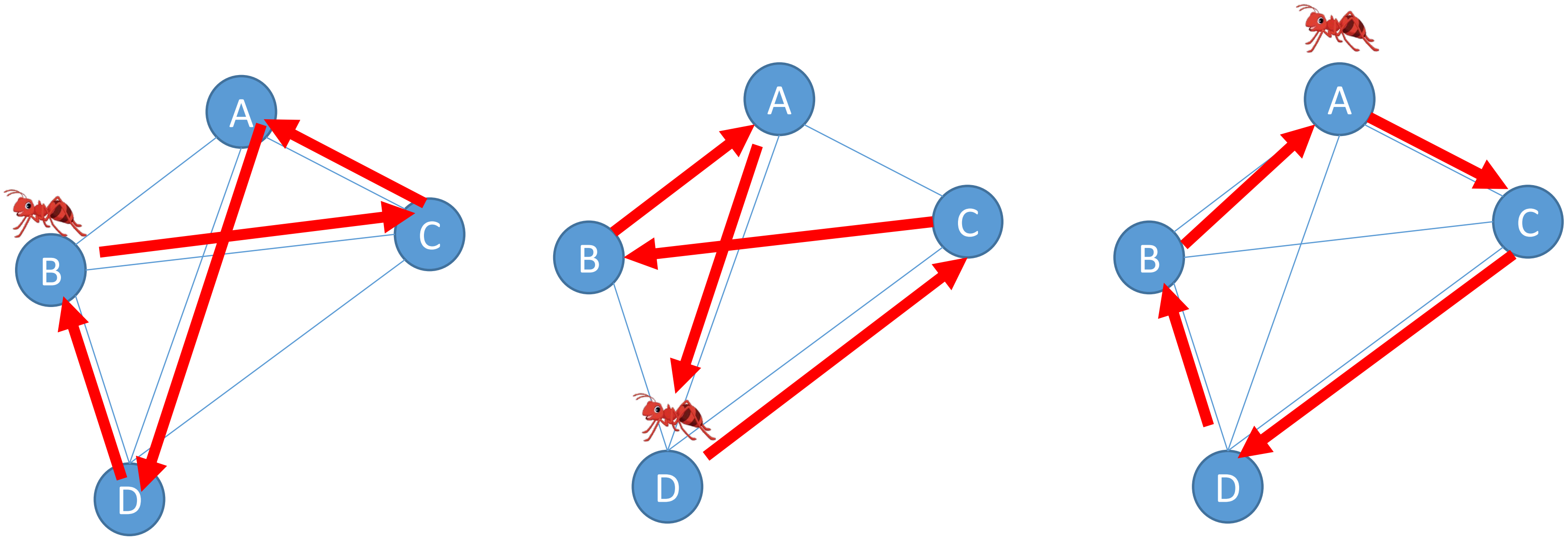
ρ: 衰减系数

m: 蚂蚁总数

$A_k(r, s)$: 如果蚂蚁k没走过rs，值为0

如果蚂蚁k走过rs，值为总路程倒数

蚁群系统

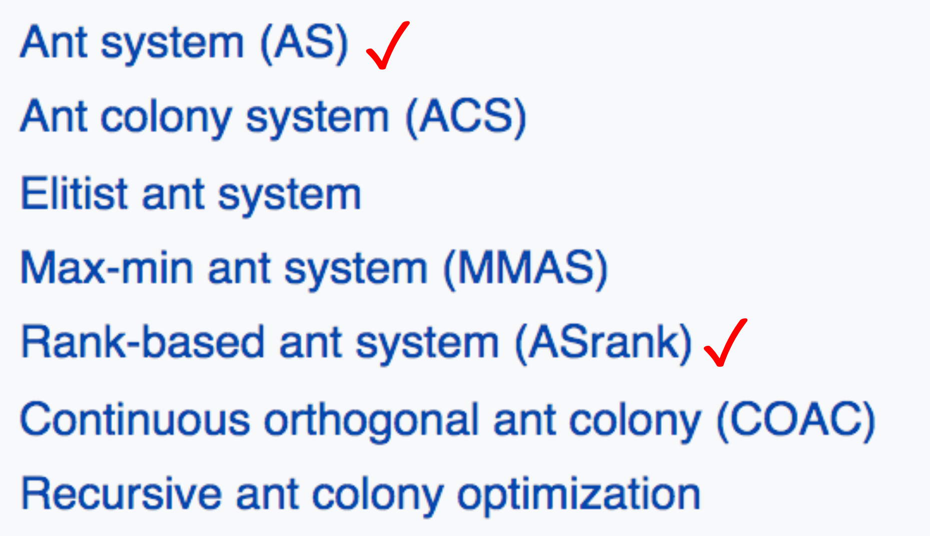


比如计算T(D, B):

$$T\left(r,s\right) \leftarrow \rho \cdot T(r,s) + \sum_{k=1}^{m} A_k(r,s)$$

$$T\left(D,B\right) \leftarrow \rho T\left(D,B\right) + \frac{1}{\mathrm{lenth}_1} + \frac{1}{\mathrm{lenth}_3}$$

蚁群优化



Ant system (AS) ✓

Ant colony system (ACS)

Elitist ant system

Max-min ant system (MMAS)

Rank-based ant system (ASrank) ✓

Continuous orthogonal ant colony (COAC)

Recursive ant colony optimization

基于排序的蚁群系统

蚁群系统（ant system）

所有m只蚂蚁都对此项贡献

$$p_k(r,s)=\frac{T(r,s)\cdot H(r,s)^{\beta}}{\sum\limits_{\text{unvisited cities }c} T(r,c)\cdot H(r,c)^{\beta}}$$

随机生成新解

$$T\left(r,s\right)\leftarrow \rho\cdot T(r,s)+\sum_{k=1}^{m}A_k(r,s)$$

更新信息素浓度

基于排序的蚁群系统（rank-based ant system）

思想: 改进更新信息素浓度方法

对蚂蚁排序，加强排序在前的蚂蚁对信息素浓度的影响

对所有蚂蚁根据路径总长由短到长排序，只允许前面若干蚂蚁对此项贡献

基于排序的蚁群系统

所有m只蚂蚁都对此项贡献

蚁群系统（ant system）

$$T\left(r,s\right)\leftarrow \rho\cdot T(r,s)+\sum_{k=1}^{m}A_k(r,s)$$

基于排序的蚁群系统（rank-based ant system）

$$T\left(r,s\right)\leftarrow \rho T\left(r,s\right)+\boxed{\sum_{R=1}^{w-1}\left(w-R\right)A^{R}\left(r,s\right)+wA^{\text{best}}\left(r,s\right)}$$

w只蚂蚁: (1) w-1只当前迭代回合所有m只蚂蚁中排前面的w-1只

(2) 历史上所有尝试过的蚂蚁中最佳的蚂蚁

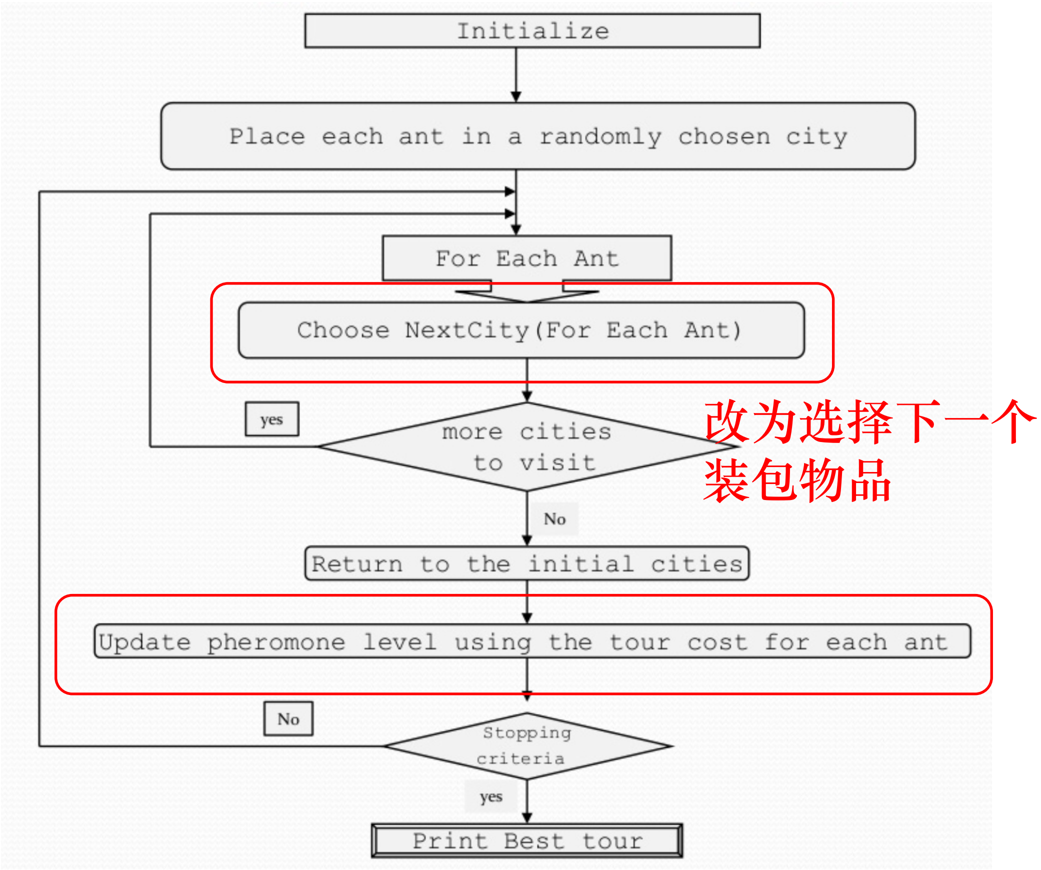
0-1背包问题

$$\max \sum_{i=1}^{n} z_i x_i$$

$$\sum_{i=1}^{n} w_i x_i \le C$$

$$x_i \in \{0,1\}, i = 1,2,\ldots,n$$

如何用类似思路求解0-1背包问题?


Initialize
Place each ant in a randomly chosen city
For Each Ant
Choose NextCity(For Each Ant)
yes
more cities to visit
No
Return to the initial cities
Update pheromone level using the tour cost for each ant
No
Stopping criteria
yes
Print Best tour

改为选择下一个装包物品

0-1背包问题

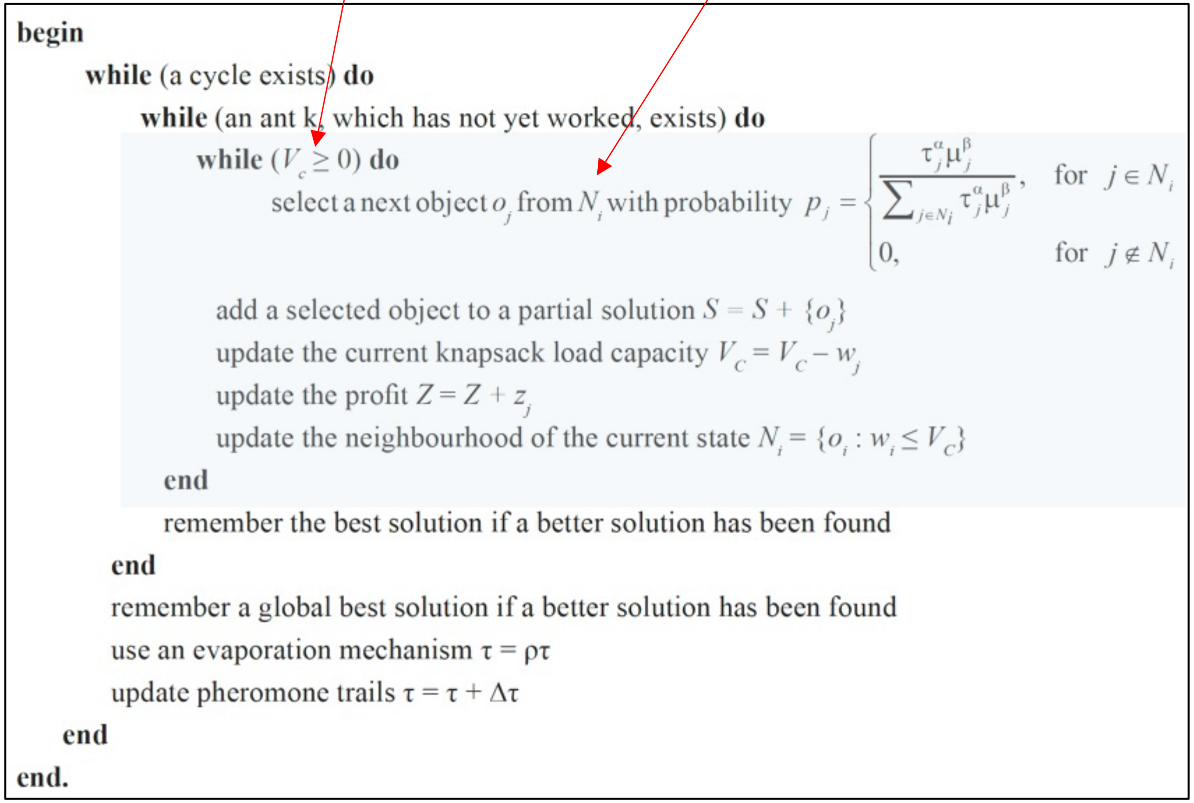
$$\max \sum_{i=1}^{n} z_i x_i$$

$$\sum_{i=1}^{n} w_i x_i \leq C$$

$$x_i \in \{0,1\}, i = 1, 2, \ldots, n$$

V_c:背包剩余空间

N_i:剩余可放入背包物品集合



```
begin
    while (a cycle exists) do
        while (an ant k, which has not yet worked, exists) do
            while ($V_c \geq 0$) do
                select a next object $o_j$ from $N_i$ with probability $p_j = \begin{cases} \frac{\tau_j^\alpha \mu_j^\beta}{\sum_{j \in N_i} \tau_j^\alpha \mu_j^\beta}, & \text{for } j \in N_i \\ 0, & \text{for } j \notin N_i \end{cases}$
              add a selected object to a partial solution $S = S + \{o_j\}$
              update the current knapsack load capacity $V_C = V_C - w_j$
              update the profit $Z = Z + z_j$
              update the neighbourhood of the current state $N_i = \{o_i : w_i \leq V_C\}$
          end
          remember the best solution if a better solution has been found
      end
      remember a global best solution if a better solution has been found
      use an evaporation mechanism $\tau = \rho\tau$
      update pheromone trails $\tau = \tau + \Delta\tau$
   end
end.
```

伪码取自Krzysztof Schiff, “Ant colony optimization algorithm for the 0-1 knapsack problem”

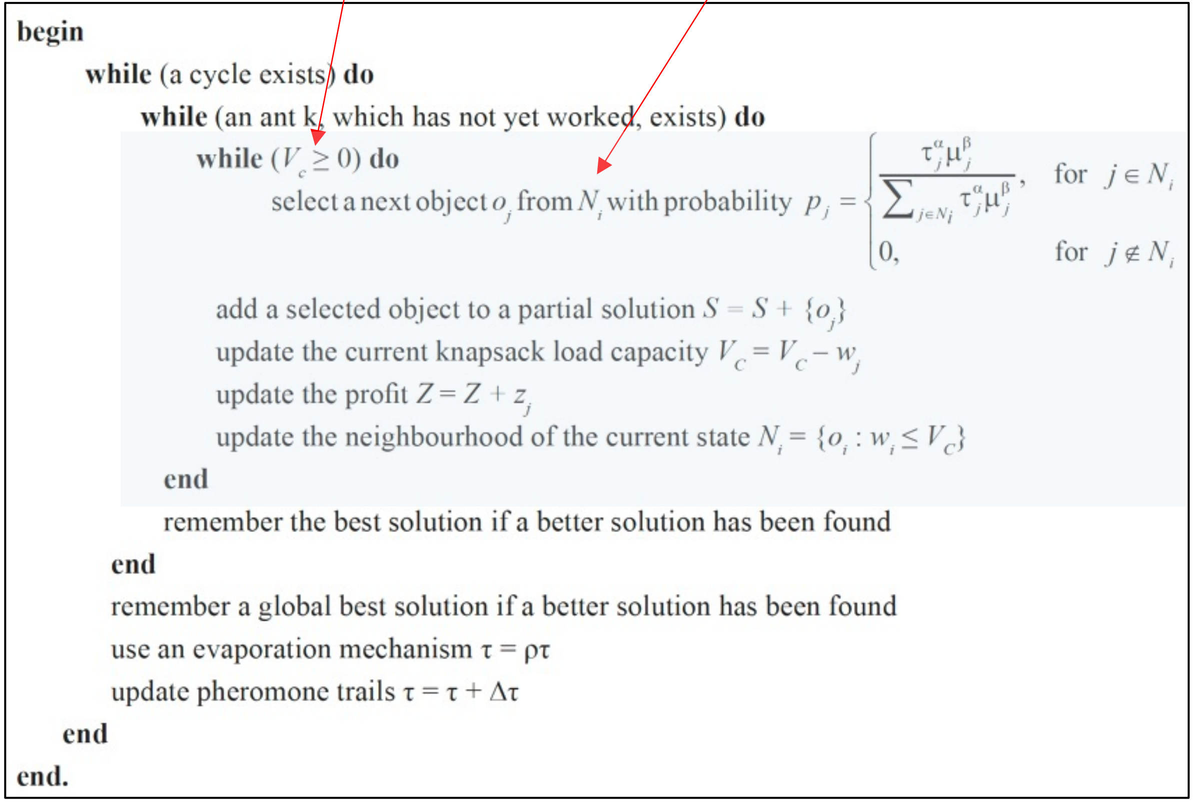
0-1背包问题

$$\max \sum_{i=1}^{n} z_i x_i$$

$$\sum_{i=1}^{n} w_i x_i \le C$$

$$x_i \in \{0,1\}, i = 1,2,\ldots,n$$

V_c:背包剩余空间　　N_i:剩余可放入背包物品集合



```
begin
    while (a cycle exists) do
        while (an ant k, which has not yet worked, exists) do
            while ($V_c \ge 0$) do
                select a next object $o_j$ from $N_i$ with probability $p_j = \begin{cases} \frac{\tau_j^\alpha \mu_j^\beta}{\sum_{j\in N_i} \tau_j^\alpha \mu_j^\beta}, & \text{for } j \in N_i \\ 0, & \text{for } j \notin N_i \end{cases}$
                add a selected object to a partial solution $S = S + \{o_j\}$
                update the current knapsack load capacity $V_c = V_c - w_j$
                update the profit $Z = Z + z_j$
                update the neighbourhood of the current state $N_i = \{o_i : w_i \le V_c\}$
            end
            remember the best solution if a better solution has been found
        end
        remember a global best solution if a better solution has been found
        use an evaporation mechanism $\tau = \rho\tau$
        update pheromone trails $\tau = \tau + \Delta\tau$
    end
end.
```

信息素浓度　　物品本身特征

$$p_j = \begin{cases} \dfrac{\tau_j^\alpha \mu_j^\beta}{\sum_{j\in N_i} \tau_j^\alpha \mu_j^\beta}, & \text{for } j \in N_i \\ 0, & \text{for } j \notin N_i \end{cases}$$

如何更新信息素浓度？如何定义物品特征μ_j？

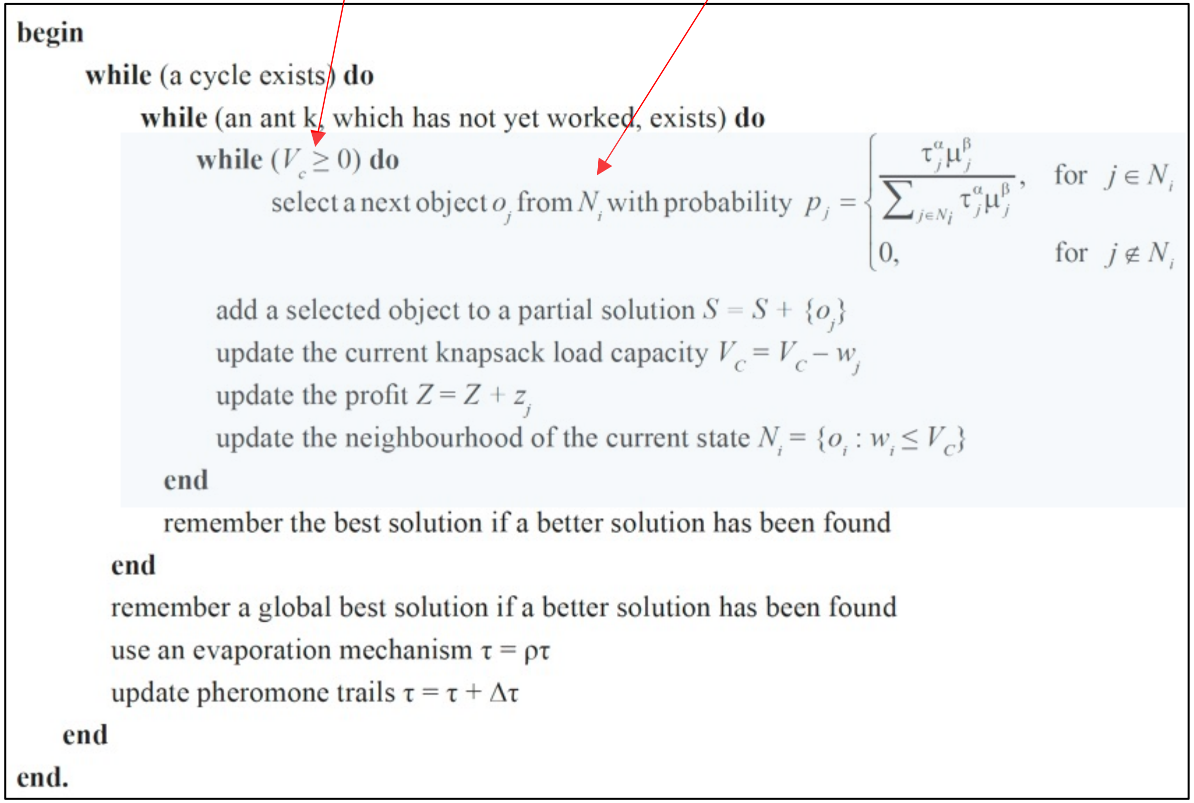
0-1背包问题

V_c:背包剩余空间 N_i:剩余可放入背包物品集合

$$\max \sum_{i=1}^{n} z_i x_i$$

$$\sum_{i=1}^{n} w_i x_i \le C$$

$$x_i \in \{0,1\}, i = 1,2,\ldots,n$$



```
begin
    while (a cycle exists) do
        while (an ant k, which has not yet worked, exists) do
            while ($V_c \ge 0$) do
                select a next object $o_j$ from $N_i$ with probability $p_j = \begin{cases} \frac{\tau_j^\alpha \mu_j^\beta}{\sum_{j\in N_i} \tau_j^\alpha \mu_j^\beta}, & \text{for } j \in N_i \\ 0, & \text{for } j \notin N_i \end{cases}$
                add a selected object to a partial solution $S = S + \{o_j\}$
                update the current knapsack load capacity $V_C = V_C - w_j$
                update the profit $Z = Z + z_j$
                update the neighbourhood of the current state $N_i = \{o_i : w_i \le V_C\}$
            end
            remember the best solution if a better solution has been found
        end
        remember a global best solution if a better solution has been found
        use an evaporation mechanism $\tau = \rho\tau$
        update pheromone trails $\tau = \tau + \Delta\tau$
    end
end.
```

信息素浓度 物品本身特征

$$p_j = \begin{cases} \frac{\tau_j^\alpha \mu_j^\beta}{\sum_{j\in N_i} \tau_j^\alpha \mu_j^\beta}, & \text{for } j \in N_i \\ 0, & \text{for } j \notin N_i \end{cases}$$

特征μ_j可以取$\frac{z_j}{w_j}$

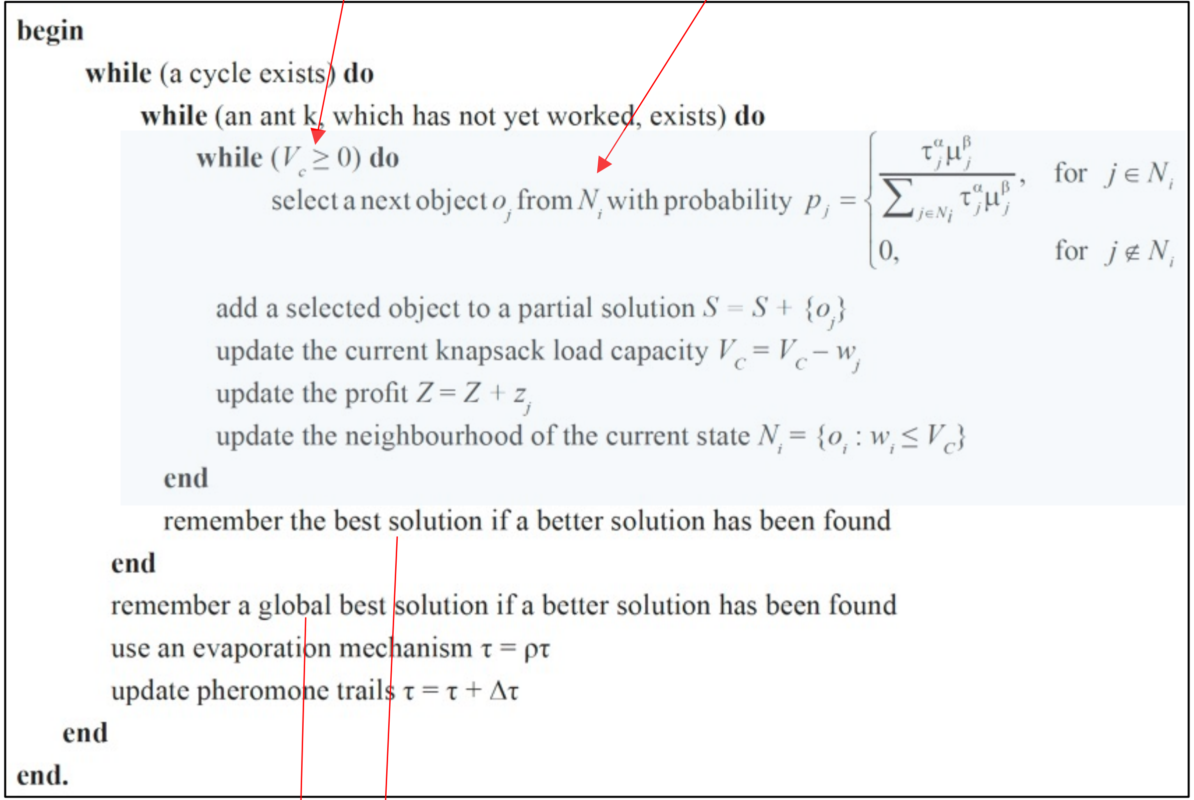
0-1背包问题

$$\max \sum_{i=1}^{n} z_i x_i$$

$$\sum_{i=1}^{n} w_i x_i \le C$$

$$x_i \in \{0,1\}, i = 1, 2, \ldots, n$$

V_c:背包剩余空间 N_i:剩余可放入背包物品集合



```
begin
    while (a cycle exists) do
        while (an ant k, which has not yet worked, exists) do
            while ($V_c \ge 0$) do
                select a next object $o_j$ from $N_i$ with probability $p_j = \begin{cases}\frac{\tau_j^{\alpha}\mu_j^{\beta}}{\sum_{j\in N_i}\tau_j^{\alpha}\mu_j^{\beta}}, & \text{for } j \in N_i \\ 0, & \text{for } j \notin N_i\end{cases}$

              add a selected object to a partial solution $S = S + \{o_j\}$
              update the current knapsack load capacity $V_c = V_c - w_j$
              update the profit $Z = Z + z_j$
              update the neighbourhood of the current state $N_i = \{o_i : w_i \le V_c\}$
            end
            remember the best solution if a better solution has been found
        end
        remember a global best solution if a better solution has been found
        use an evaporation mechanism $\tau = \rho\tau$
        update pheromone trails $\tau = \tau + \Delta\tau$
    end
end.
```

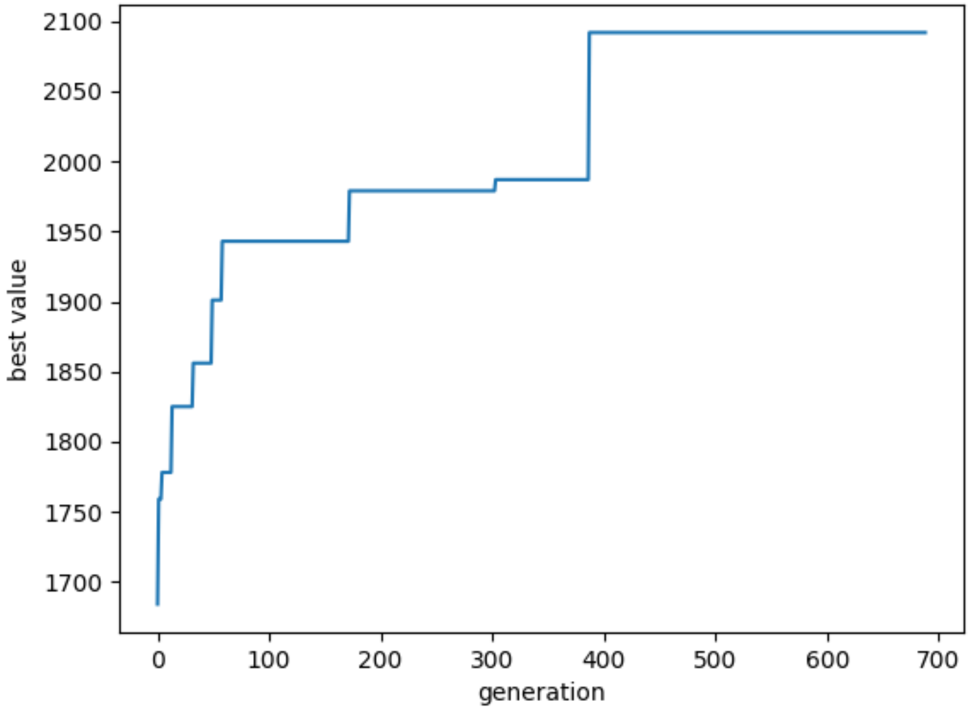
信息素浓度 物品本身特征

$$p_j = \begin{cases}\frac{\tau_j^{\alpha}\mu_j^{\beta}}{\sum_{j\in N_i}\tau_j^{\alpha}\mu_j^{\beta}}, & \text{for } j \in N_i \\ 0, & \text{for } j \notin N_i\end{cases}$$

$$\Delta\tau_j\text{可以取}\begin{cases}0, & \text{若当前迭代回合最佳方案不含物品j,} \\ \frac{1}{1+\frac{z_{best}-z}{z_{best}}}, & \text{若当前迭代回合最佳方案含物品j.}\end{cases}$$

0-1背包问题

n=100, C=1000 代码结果示例:



群体智能之粒子群优化

Part1 如何从复杂度的角度衡量算法好坏？

Part2 针对一些有特定形式的最优化问题，如何找到最优解？

Part3 针对较复杂的最优化问题（如NP难问题），如何找到较好解？

局部搜索、模拟退火、遗传算法、差分进化算法、蚁群算法、粒子群优化算法、人工蜂群算法、贝叶斯优化

Part4 针对较复杂的多目标最优化问题，如何找到较好解？

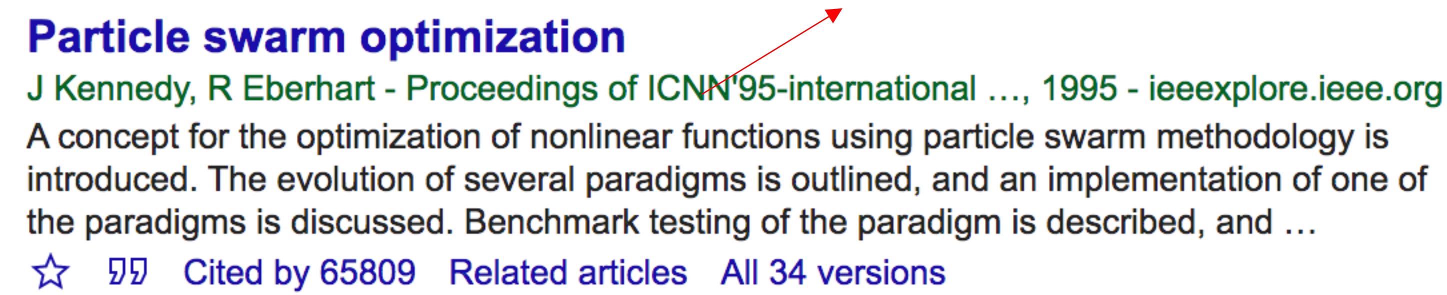
粒子群优化

粒子群优化（Particle Swarm Optimization）

受鸟群、鱼群的群体活动启发而设计

粒子群优化

International Conference on Neural Networks



Particle swarm optimization

J Kennedy, R Eberhart - Proceedings of ICNN'95-international …, 1995 - ieeexplore.ieee.org

A concept for the optimization of nonlinear functions using particle swarm methodology is introduced. The evolution of several paradigms is outlined, and an implementation of one of the paradigms is discussed. Benchmark testing of the paradigm is described, and …

☆ ❞ Cited by 65809 Related articles All 34 versions

思想：将个体经验和群体经验结合

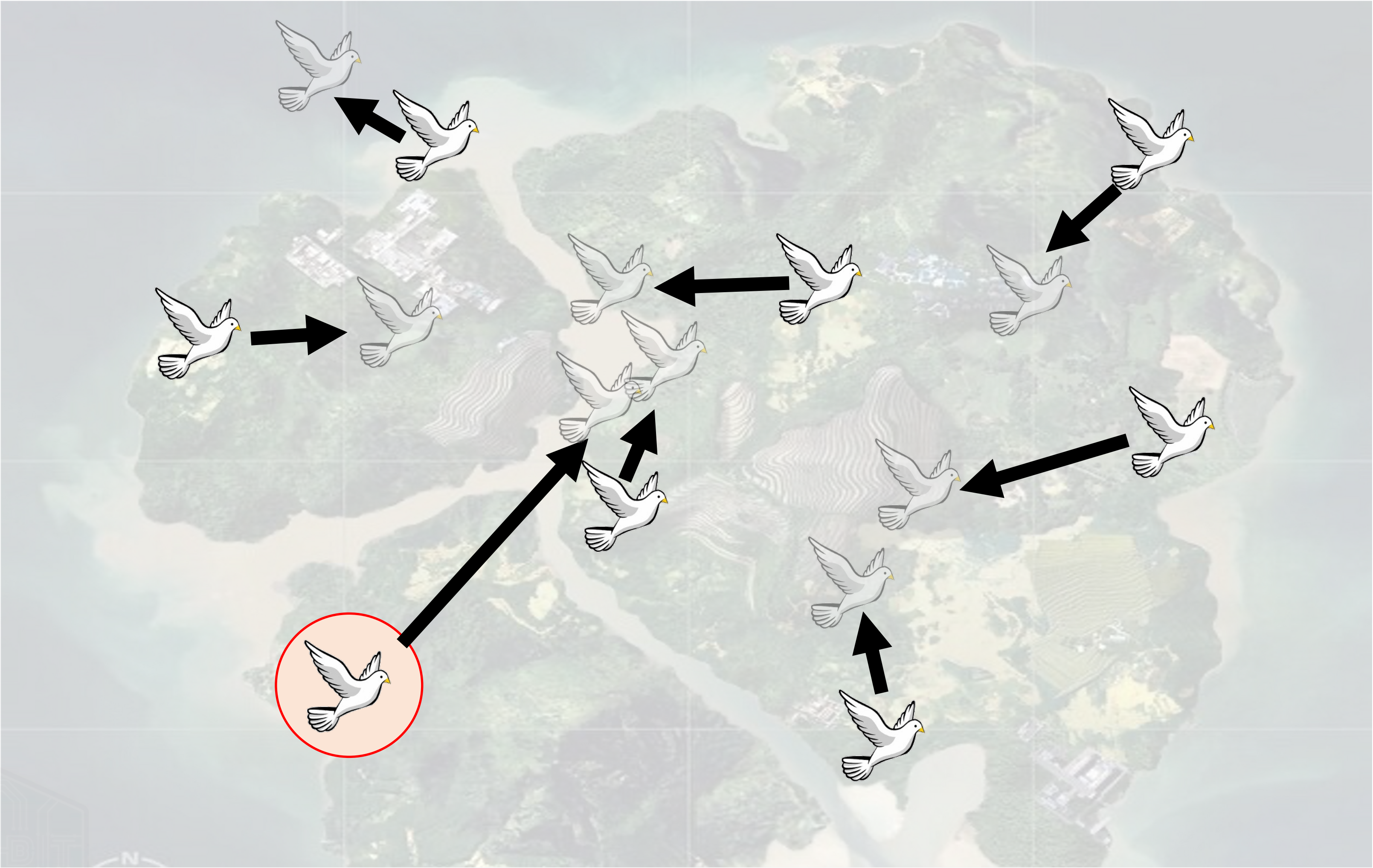
主要思想

二维连续空间找到“食物最多”的地点

主要思想

如何确定每只鸟当前的“速度”？

即下一时刻位置与当前时刻位置的差值（即一个向量）



主要思想

考虑三方面的因素：

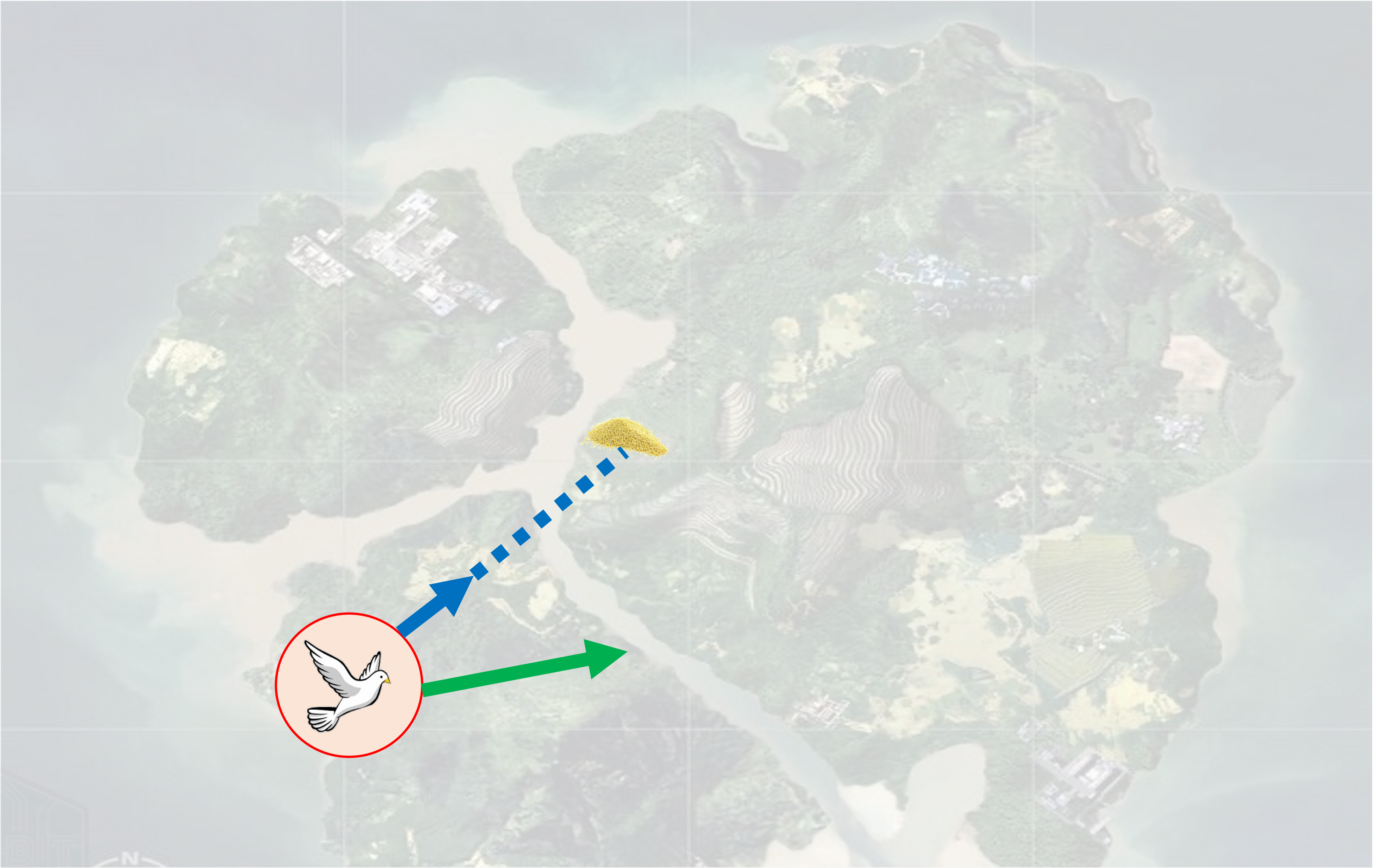
（1）上一时刻“速度”

主要思想

考虑三方面的因素：

（1）上一时刻“速度”

（2）该只鸟在历史中找到的最佳取食点的方向



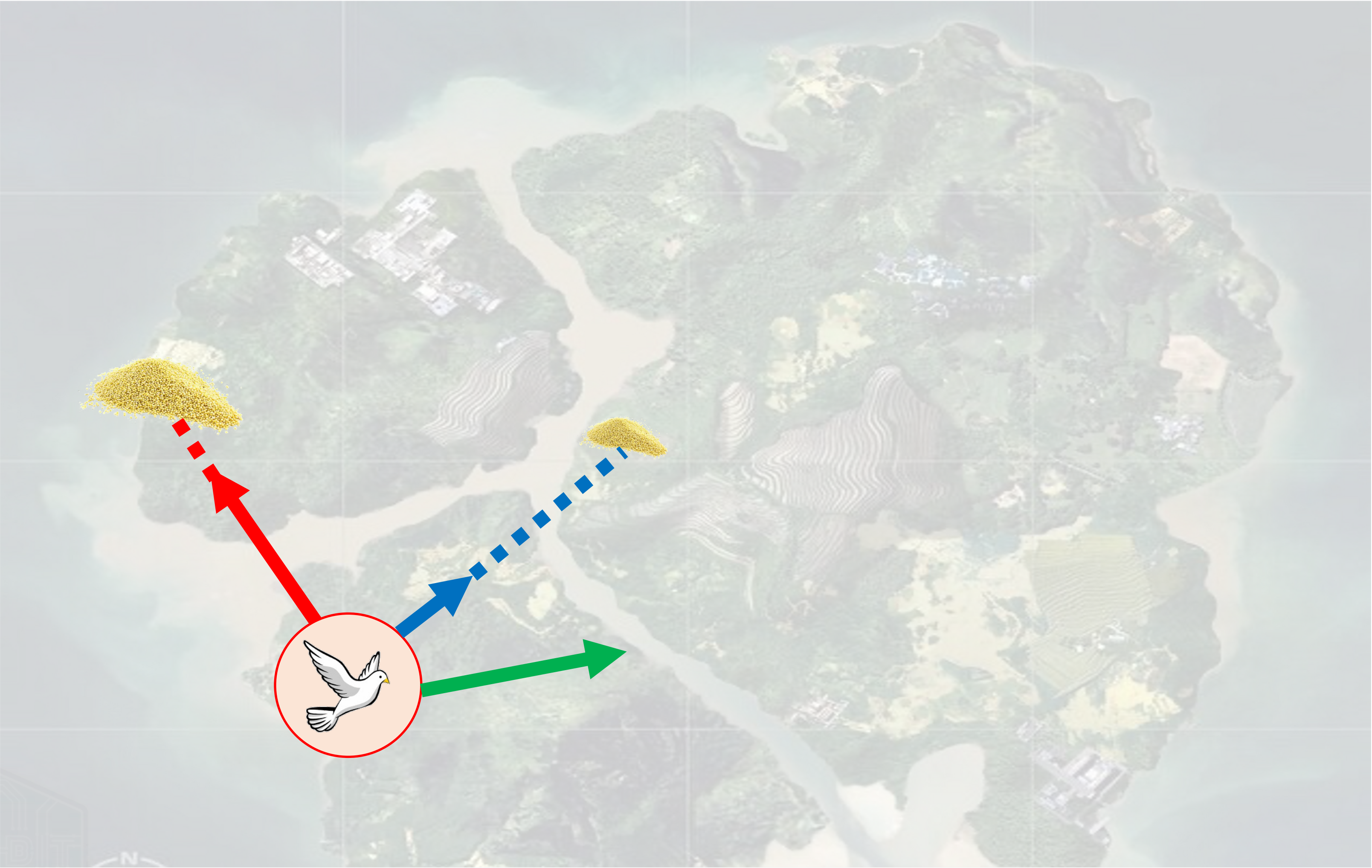
主要思想

考虑三方面的因素：

（1）上一时刻“速度”

（2）该只鸟在历史中找到的最佳取食点的方向

（3）所有鸟在历史中找到的最佳取食点的方向



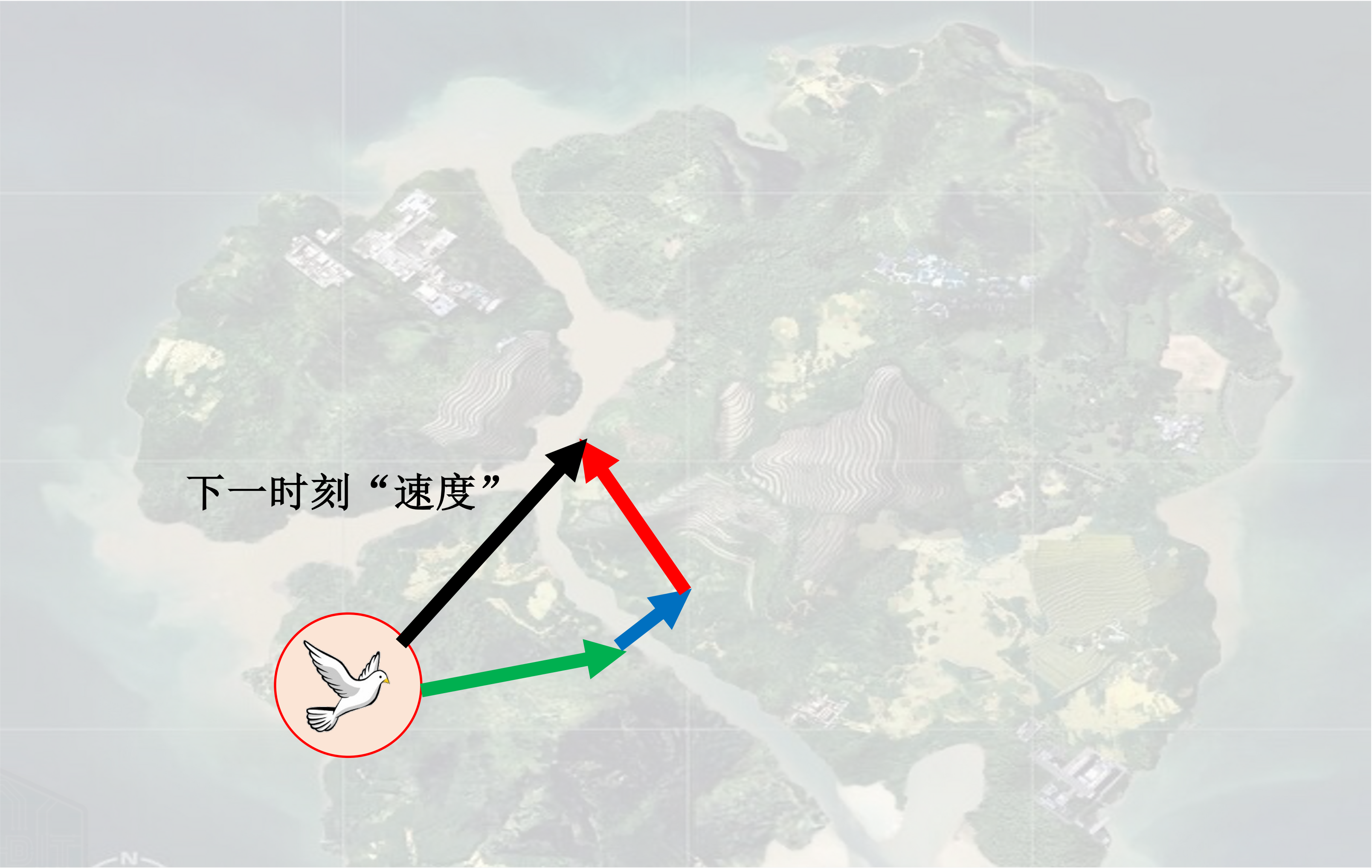
主要思想

考虑三方面的因素：

（1）上一时刻“速度”

（2）该只鸟在历史中找到的最佳取食点的方向

（3）所有鸟在历史中找到的最佳取食点的方向



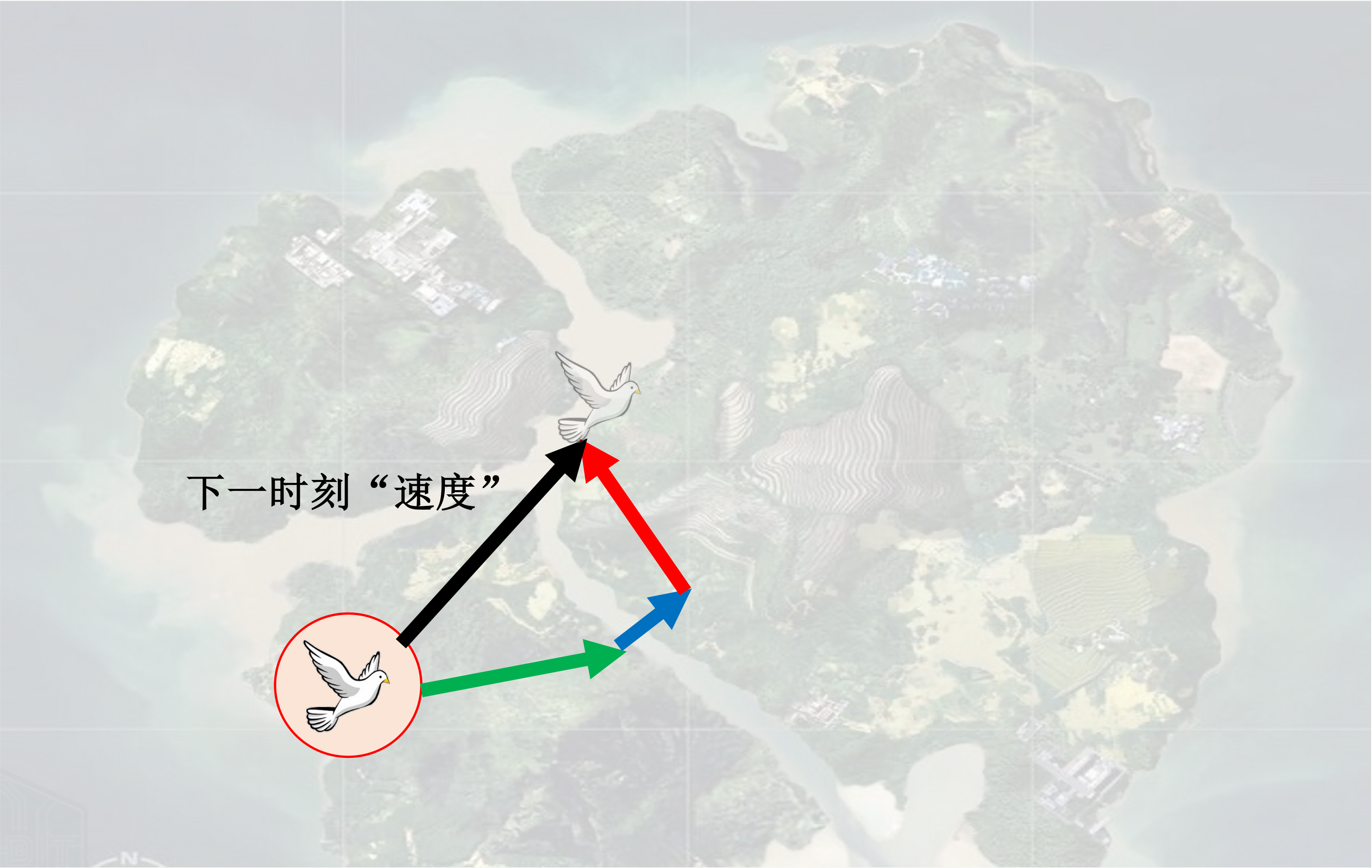
主要思想

考虑三方面的因素：

（1）上一时刻“速度”

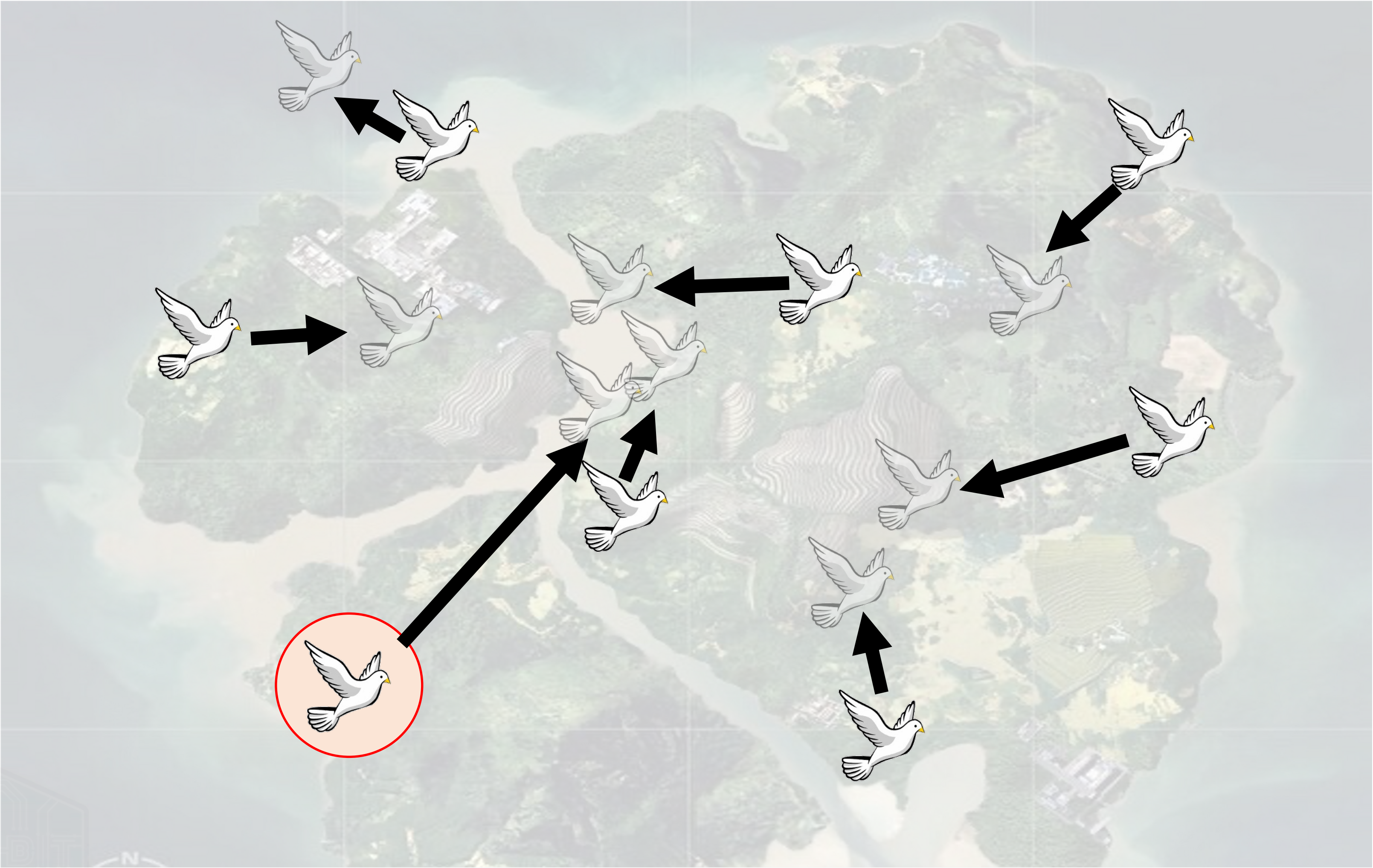
（2）该只鸟在历史中找到的最佳取食点的方向

（3）所有鸟在历史中找到的最佳取食点的方向

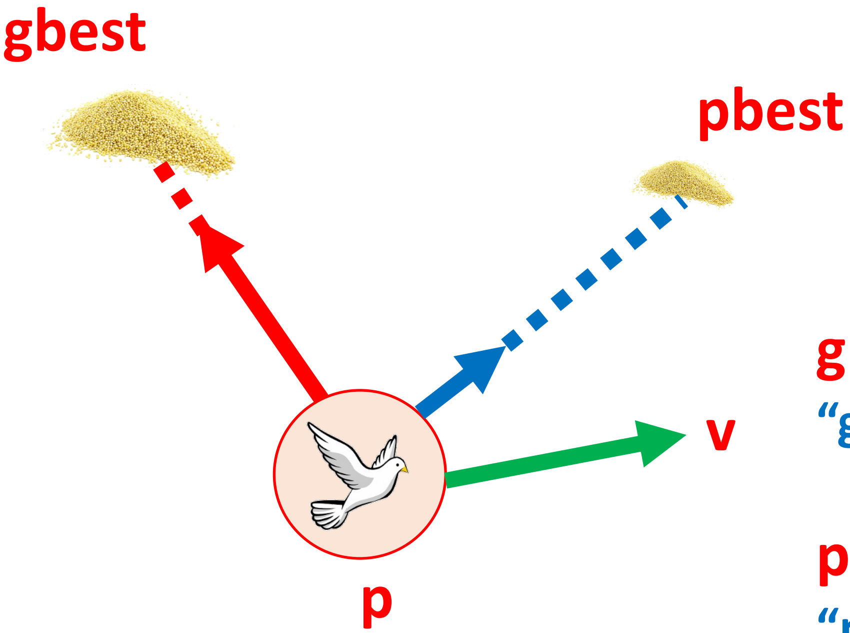


主要思想

所有鸟分别根据三因素计算当前“速度”
根据“速度”移动到下一位置



主要思想



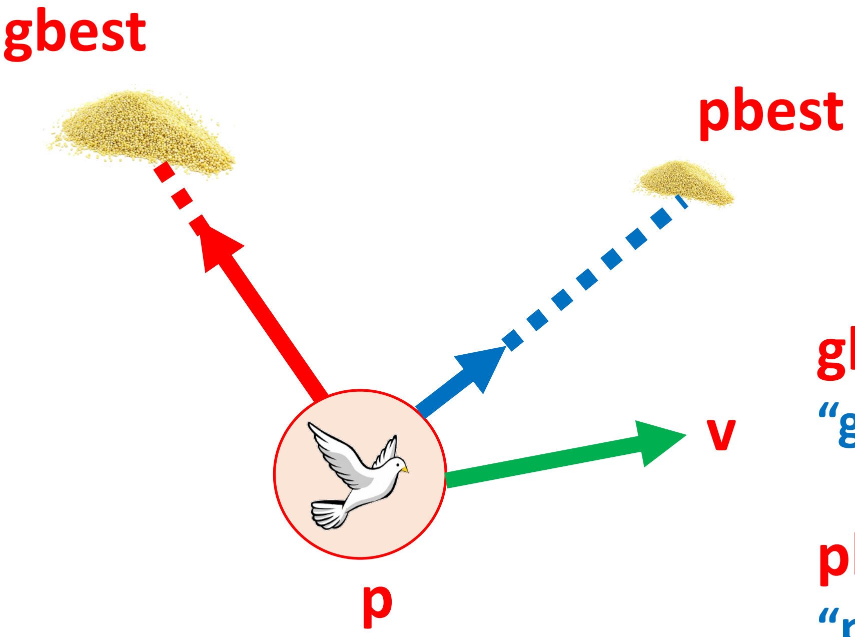
p: 该只鸟当前位置

v: 该只鸟上一时刻速度

gbest: 所有鸟在历史中找到的最佳取食点
“g” – group

pbest: 该只鸟在历史中找到的最佳取食点
“p” – particle (粒子)

主要思想



p: 该只鸟当前位置

v: 该只鸟上一时刻速度

gbest: 所有鸟在历史中找到的最佳取食点
"g" – group

pbest: 该只鸟在历史中找到的最佳取食点
"p" – particle (粒子)

下一时刻速度计算方法：

$v \leftarrow w*v + c1*rand*(\underline{pBest - p}) + c2*rand*(\underline{gBest - p})$

粒子群优化

```
[x*] = PSO()
$P$ = Particle_Initialization();
For $i$=1 to $it\_max$          确定迭代次数
    For each particle $p$ in $P$ do
        $fp$ = f($p$);
        If $fp$ is better than f($pBest$)          对每只鸟更新pBest
            $pBest = p$;
        end
    end
    $gBest$ = best $pBest$ in $P$;          根据已更新的pBest，更新gBest
    For each particle $p$ in $P$ do
        $v = w*v + c1*rand*(pBest - p) + c2*rand*(gBest - p)$;  对每只鸟，更新速度
        $p = p + v$;                                            根据速度，更新位置
    end
end
```

最后输出是*gBest*

粒子群优化

下一时刻速度计算方法：

$$v \leftarrow w*v + c1*rand*(pBest - p) + c2*rand*(gBest - p)$$

1. Inertia

2. Personal Influence

3. Social Influence

思考: 三部分中哪些对应exploration、哪些对应exploitation？

例子

考虑最优化问题 $$\min \quad f(x)=\sum_{i=1}^{4} x_i^2; \quad 0 \le x_i \le 10, \quad i=1,2,3,4$$

第一步 确定超参数（种群大小、惯性系数、速度计算系数、迭代回合数）

$$N_P = 5,\ w = 0.7,\ c_1 = 1.5,\ c_2 = 1.5,\ T = 10$$

第二步 在可行域内随机生成初代种群，计算解的质量

$$P = \begin{bmatrix} 4 & 0 & 0 & 8 \\ 3 & 1 & 9 & 7 \\ 0 & 3 & 1 & 5 \\ 2 & 1 & 4 & 9 \\ 6 & 2 & 8 & 3 \end{bmatrix} \qquad f = \begin{bmatrix} 80 \\ 140 \\ 35 \\ 102 \\ 113 \end{bmatrix}$$

例子及求解过程修改自Prakash Kotecha (IIT Guwahati) 课程 Computer Aided Applied Single Objective Optimization

例子

第三步 随机生成速度向量

$$P=\begin{bmatrix}4 & 0 & 0 & 8\\ 3 & 1 & 9 & 7\\ 0 & 3 & 1 & 5\\ 2 & 1 & 4 & 9\\ 6 & 2 & 8 & 3\end{bmatrix} \qquad f=\begin{bmatrix}80\\ 140\\ 35\\ 102\\ 113\end{bmatrix} \qquad v=\begin{bmatrix}9 & 6 & 1 & 8\\ 5 & 1 & 3 & 0\\ 7 & 4 & 1 & 4\\ 3 & 0 & 2 & 1\\ 1 & 6 & 8 & 7\end{bmatrix}$$

第四步 确定当前的pBest和gBest

$$P_{best}=P=\begin{bmatrix}4 & 0 & 0 & 8\\ 3 & 1 & 9 & 7\\ 0 & 3 & 1 & 5\\ 2 & 1 & 4 & 9\\ 6 & 2 & 8 & 3\end{bmatrix} \qquad f_{pbest}=\begin{bmatrix}80\\ 140\\ 35\\ 102\\ 113\end{bmatrix} \qquad g_{best}=\begin{bmatrix}0 & 3 & 1 & 5\end{bmatrix} \qquad f_{gbest}=35$$

例子

针对第一个粒子

$$w = 0.7,\ c_1 = 1.5,\ c_2 = 1.5,\ T = 10$$

$$v_1 = \begin{bmatrix}9 & 6 & 1 & 8\end{bmatrix}$$

$$X_1 = \begin{bmatrix}4 & 0 & 0 & 8\end{bmatrix} \qquad f = 80$$

$$p_{best,1} = \begin{bmatrix}4 & 0 & 0 & 8\end{bmatrix} \qquad f_{pbest_1} = 80$$

$$gbest = \begin{bmatrix}0 & 3 & 1 & 5\end{bmatrix} \qquad f_{gbest} = 35$$

第五步 随机生成r1和r2

$$r_1 = [0.4\ \ 0.3\ \ 0.9\ \ 0.5\,] \qquad r_2 = [0.8\ \ 0.2\ \ 0.7\ \ 0.4]$$

第六步 根据公式计算新的速度向量

$$v_i = wv_i + c_1 r_1 \left(p_{best,i} - X_i\right) + c_2 r_2 \left(g_{best} - X_i\right)$$

$$X_i = X_i + v_i$$

$$\begin{aligned} v_1 &= 0.7 \text{ x } [9\ \ 6\ \ 1\ \ 8] + \\ &\quad 1.5 \text{ x } [0.4\ \ 0.3\ \ 0.9\ \ 0.5\,] \text{ x } ([4\ 0\ 0\ 8] - [4\ 0\ 0\ 8]) + \\ &\quad 1.5 \text{ x } [0.8\ \ 0.2\ \ 0.7\ \ 0.4] \text{ x } ([0\ 3\ 1\ 5] - [4\ 0\ 0\ 8]) \\ v_1 &= [1.5\ \ 5.1\ \ 1.75\ \ 3.8] \end{aligned}$$

第七步 得到该粒子的新的位置

$$\begin{aligned} X_1 &= [4\ 0\ 0\ 8] + [1.5\ \ 5.1\ \ 1.75\ \ 3.8] \\ &= [5.5\ \ 5.1\ \ 1.75\ \ 11.8] \end{aligned}$$

例子

针对第一个粒子

第八步 确保粒子的新位置在可行域内 $\boxed{0 \le x_i \le 10}$

$$X_1 = [5.5 \quad 5.1 \quad 1.75 \quad 11.8] \qquad X_1 = [5.5 \quad 5.1 \quad 1.75 \quad 10]$$

第九步 为修改后的新解计算它的质量

$$f_1 = 5.5^2 + 5.1^2 + 1.75^2 + 10^2 = 159.32$$

第十步 更新种群

$$Pop = \begin{bmatrix} 5.5 & 5.1 & 1.75 & 10 \\ 3 & 1 & 9 & 7 \\ 0 & 3 & 1 & 5 \\ 2 & 1 & 4 & 9 \\ 6 & 2 & 8 & 3 \end{bmatrix} \qquad f = \begin{bmatrix} 159.32 \\ 140 \\ 35 \\ 102 \\ 113 \end{bmatrix}$$

第十一步 判断是否更新pBest,1和gBest

$$p_{best,1} = [4 \quad 0 \quad 0 \quad 8] \quad f_{pbest_1} = 80$$

$$g_{best} = [0 \quad 3 \quad 1 \quad 5] \quad f_{gbest} = 35$$

例子

完成第一次迭代

$$P=\begin{bmatrix}4&0&0&8\\3&1&9&7\\0&3&1&5\\2&1&4&9\\6&2&8&3\end{bmatrix}\quad f=\begin{bmatrix}80\\140\\35\\102\\113\end{bmatrix}\quad P_{best}=\begin{bmatrix}4&0&0&8\\3&1&9&7\\0&3&1&5\\2&1&4&9\\6&2&8&3\end{bmatrix}\quad f_{pbest}=\begin{bmatrix}80\\140\\35\\102\\113\end{bmatrix}\quad v=\begin{bmatrix}9&6&1&8\\5&1&3&0\\7&4&1&4\\3&0&2&1\\1&6&8&7\end{bmatrix}$$

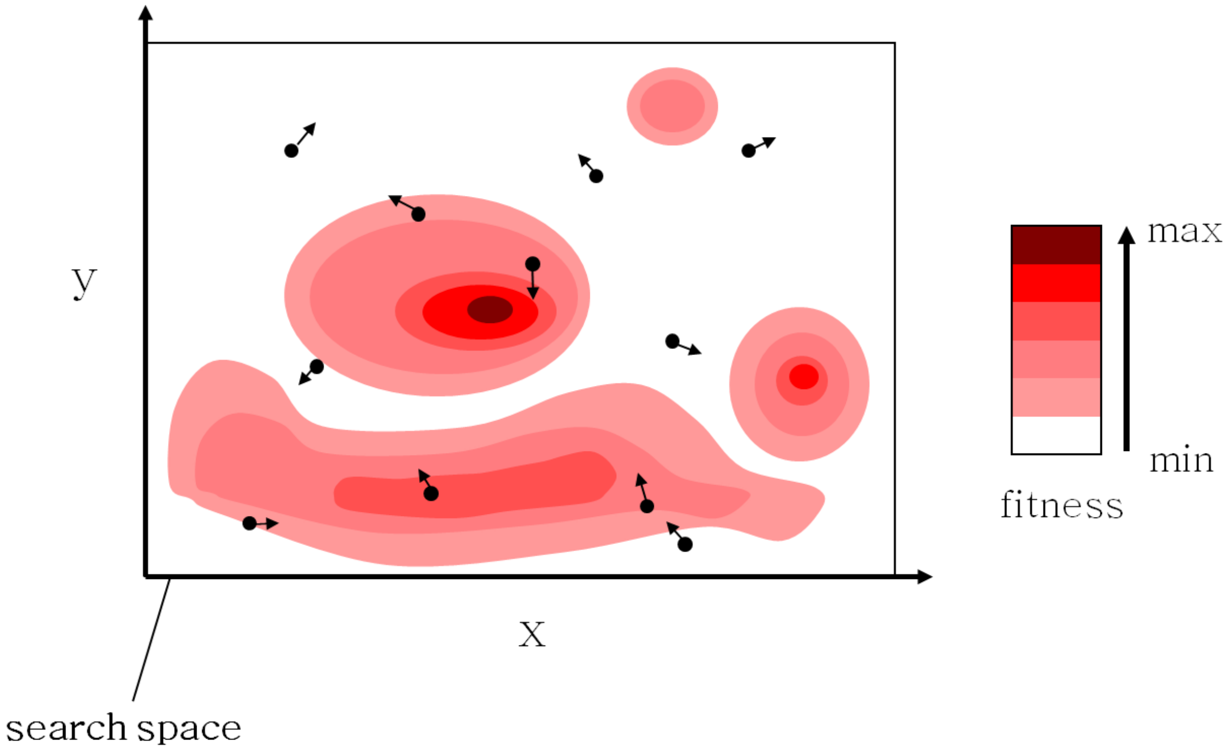
$$g_{best}=\begin{bmatrix}0&3&1&5\end{bmatrix}$$

$$f_{gbest}=35$$

$$P=\begin{bmatrix}5.5&5.1&1.75&10\\3.35&3.2&1.5&6.4\\4.9&5.8&1.7&7.8\\1.7&1.3&2.25&4.3\\3.48&6.09&10&9.26\end{bmatrix}\quad f=\begin{bmatrix}159.32\\64.67\\121.38\\28.13\\235.95\end{bmatrix}\quad P_{best}=\begin{bmatrix}4&0&0&8\\3.35&3.2&1.5&6.4\\0&3&1&5\\1.7&1.3&2.25&4.3\\6&2&8&3\end{bmatrix}\quad f_{pbest}=\begin{bmatrix}80\\64.67\\35\\28.13\\113\end{bmatrix}\quad v=\begin{bmatrix}1.5&5.1&1.75&3.8\\0.35&2.2&-7.5&-0.6\\4.9&2.8&0.7&2.8\\-0.3&0.3&-1.75&-4.7\\-2.52&4.09&3.87&6.26\end{bmatrix}$$

$$g_{best}=\begin{bmatrix}1.7&1.3&2.25&4.3\end{bmatrix},\ f_{gbest}=28.13$$

例子



粒子群算法可视化

https://www.bilibili.com/video/BV1Lf4y1v7pt?from=search&seid=12959864526843471690

粒子群优化

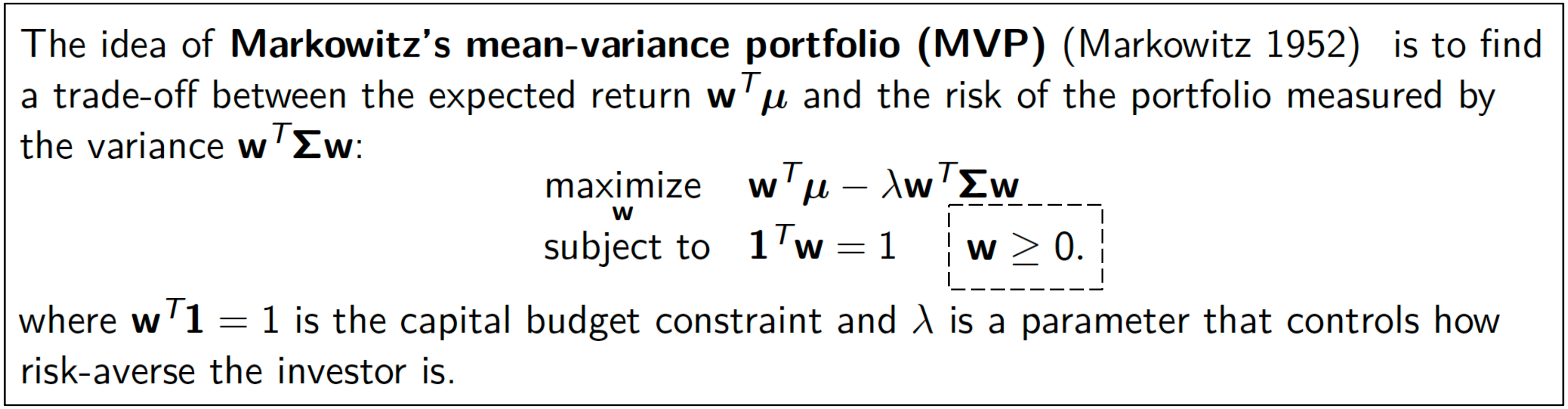
- 优势
  - 易于实施
  - 易于并行运行
  - 算法参数少
  - 全局搜索性能好

- 劣势
  - 易于提早收敛到次优解
  - 在局部精确搜索能力差 （可以将PSO与local search结合）

投资组合优化

如何对不同资产进行投资，从而最大化收益的期望、最小化风险（方差）

投资策略记为$\boldsymbol{w} = (w_1, w_2, \dots, w_N)^T$



The idea of **Markowitz's mean-variance portfolio (MVP)** (Markowitz 1952) is to find a trade-off between the expected return $\mathbf{w}^T\boldsymbol{\mu}$ and the risk of the portfolio measured by the variance $\mathbf{w}^T\boldsymbol{\Sigma}\mathbf{w}$:

$$\begin{array}{ll} \underset{\mathbf{w}}{\text{maximize}} & \mathbf{w}^T\boldsymbol{\mu} - \lambda \mathbf{w}^T\boldsymbol{\Sigma}\mathbf{w} \\ \text{subject to} & \mathbf{1}^T\mathbf{w} = 1 \quad \boxed{\mathbf{w} \geq 0.} \end{array}$$

where $\mathbf{w}^T\mathbf{1} = 1$ is the capital budget constraint and λ is a parameter that controls how risk-averse the investor is.

该问题是有约束的凸优化问题

投资组合优化

如何对不同资产进行投资，从而最大化收益的期望、最小化风险（方差）

投资策略记为$\boldsymbol{w} = (w_1, w_2, \dots, w_N)^T$

给定参数

存在其它建模形式

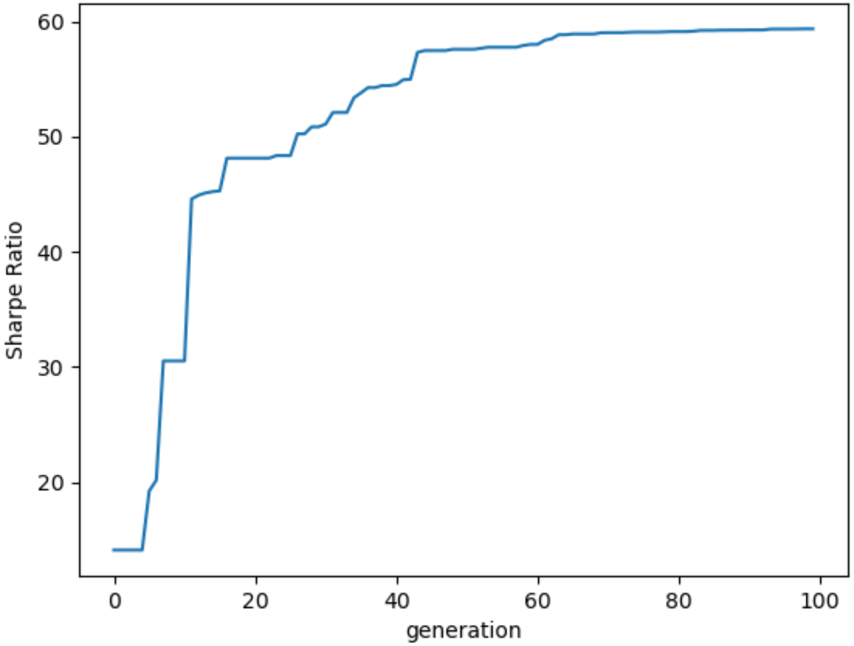
$$\text{maximize} \; \frac{\boldsymbol{w}^T\boldsymbol{\mu} - R_f}{\boldsymbol{w}^T\boldsymbol{\Sigma}\boldsymbol{w}}$$

$$\text{subject to} \quad \mathbf{1}^T\mathbf{w} = 1 \qquad \mathbf{w} \geq 0.$$

投资组合优化

8组资产，PSO算法代码结果示例：

```
[x*] = PSO()
$P$ = Particle_Initialization();
For $i$=1 to $it\_max$
    For each particle $p$ in $P$ do
        $fp = f(p)$;
        If $fp$ is better than $f(pBest)$
            $pBest = p$;
        end
    end
    $gBest$ = best $pBest$ in $P$;
    For each particle $p$ in $P$ do
        $v = w*v + c1*rand*(pBest - p)$
                $+ c2*rand*(gBest - p)$;
        $p = p + v$;
    end
end
```



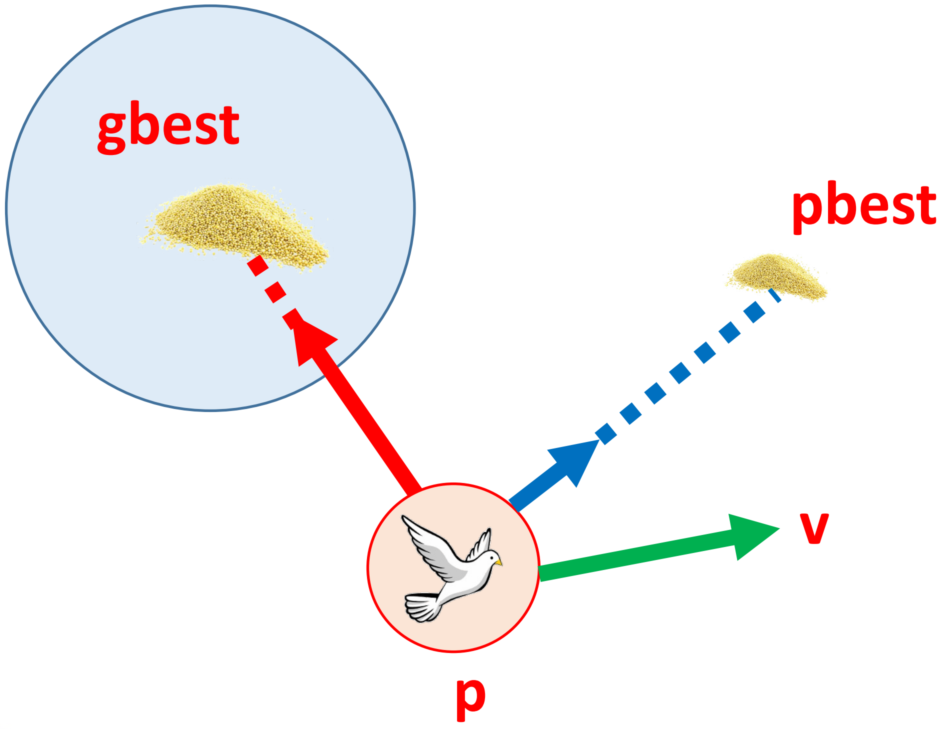
变式之一：由gBest改成lBest

```
[x*] = PSO()
$P$ = Particle_Initialization();
For $i$=1 to $it\_max$
    For each particle $p$ in $P$ do
        $fp$ = f($p$);
        If $fp$ is better than f($pBest$)
            $pBest = p$;
        end
    end
    $gBest$ = best $pBest$ in $P$;
    For each particle $p$ in $P$ do
        $v = w*v + c1*rand*(pBest - p) + c2*rand*(gBest - p)$;
        $p = p + v$;
    end
end
```

把gBest替换成每个particle独有的lbest

变式之一：由gBest改成lBest

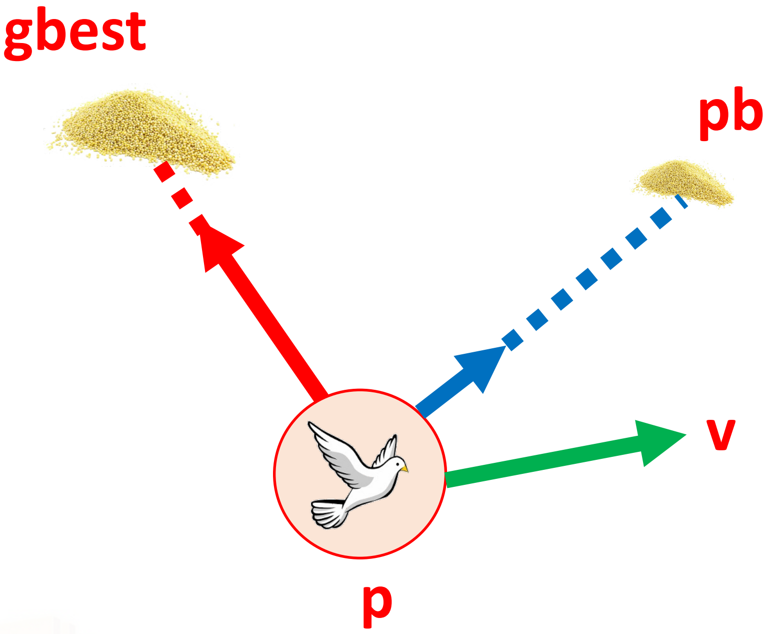
把gBest替换成每个particle独有的lbest

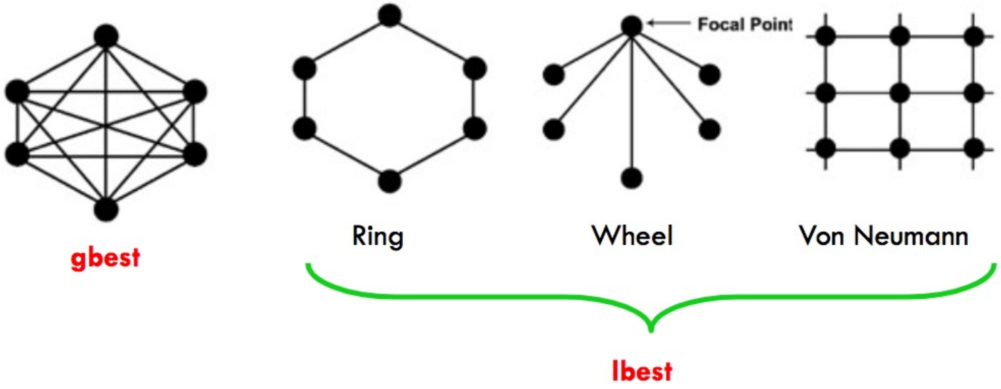


变式之一：由gBest改成lBest

把gBest替换成每个particle独有的lbest

neighbour的定义：根据解空间远近或者人为定义联系





变式之二：离散变量

```
[x*] = PSO()
P = Particle_Initialization();
For i=1 to it_max
    For each particle p in P do
        fp = f(p);
        If fp is better than f(pBest)
            pBest = p;
        end
    end
    gBest = best pBest in P;
    For each particle p in P do
        v = w*v + c1*rand*(pBest – p) + c2*rand*(gBest – p);
        p = p + v;
    end
end
```

比如求解旅行商问题，

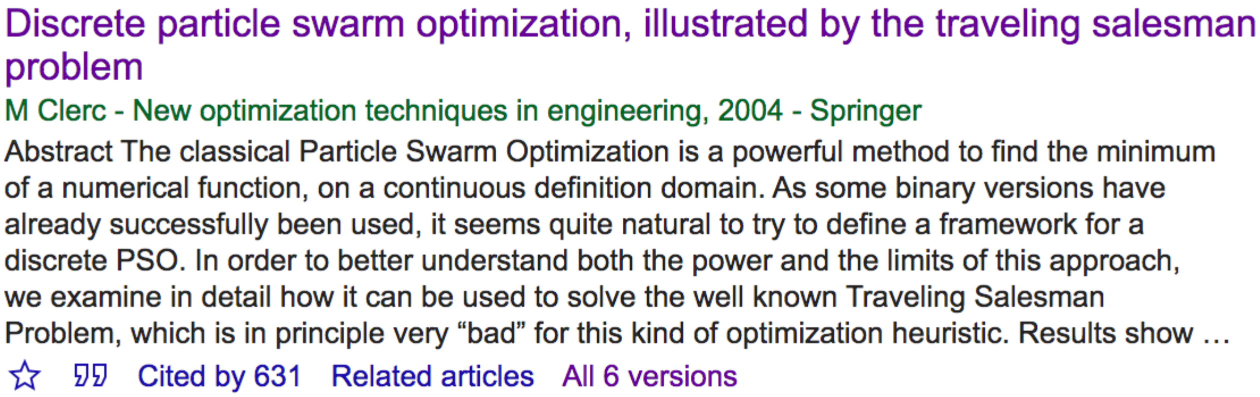
如何定义粒子？

哪些步骤需要调整？

变式之二：离散变量

$$v \leftarrow w*v + c1*rand*(pBest - p) + c2*rand*(gBest - p)$$

需要重新定义 +、*、 – 等运算



Discrete particle swarm optimization, illustrated by the traveling salesman problem

M Clerc - New optimization techniques in engineering, 2004 - Springer

Abstract The classical Particle Swarm Optimization is a powerful method to find the minimum of a numerical function, on a continuous definition domain. As some binary versions have already successfully been used, it seems quite natural to try to define a framework for a discrete PSO. In order to better understand both the power and the limits of this approach, we examine in detail how it can be used to solve the well known Traveling Salesman Problem, which is in principle very “bad” for this kind of optimization heuristic. Results show …

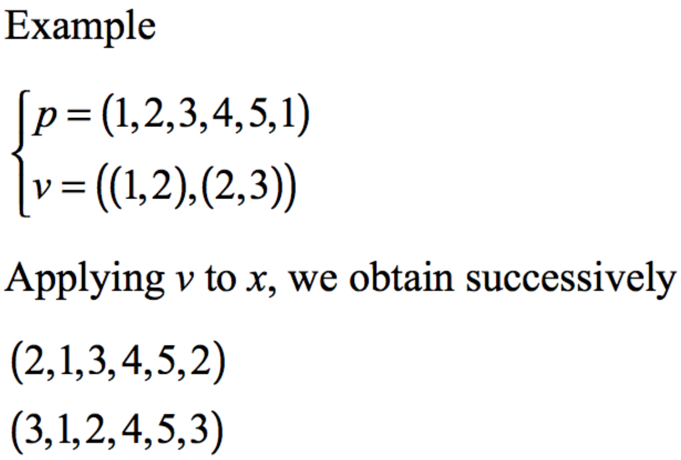
☆ ❞ Cited by 631 Related articles All 6 versions

变式之二：离散变量

$$v \leftarrow w*v + c1*rand*(pBest - p) + c2*rand*(gBest - p)$$

需要重新定义 +、*、 – 等运算

重新定义的加法运算: 粒子位置 加 速度



Example

$$\begin{cases} p = (1,2,3,4,5,1) \\ v = ((1,2),(2,3)) \end{cases}$$

Applying v to x, we obtain successively

(2,1,3,4,5,2)

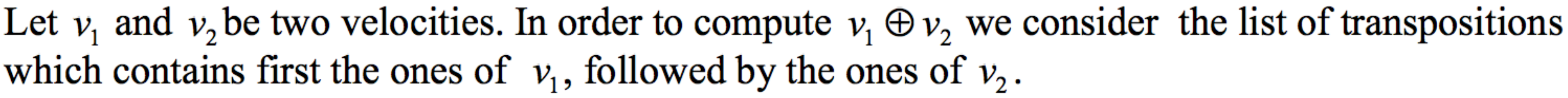
(3,1,2,4,5,3)

变式之二：离散变量

$$v \leftarrow w*v + c1*rand*(pBest - p) + c2*rand*(gBest - p)$$

需要重新定义 +、*、 – 等运算

重新定义的加法运算: 速度 加 速度



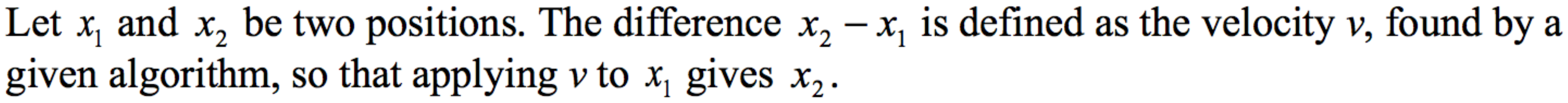
Let v_1 and v_2 be two velocities. In order to compute $v_1 \oplus v_2$ we consider the list of transpositions which contains first the ones of v_1, followed by the ones of v_2.

变式之二：离散变量

$v \leftarrow w*v + c1*rand*(pBest - p) + c2*rand*(gBest - p)$

需要重新定义 +、*、 – 等运算

重新定义的减法运算: 位置 减 位置



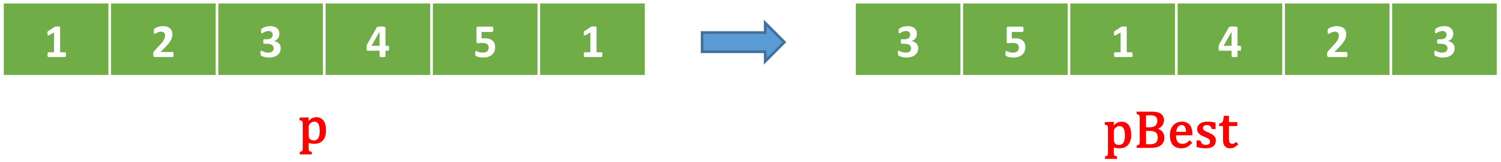
Let x_1 and x_2 be two positions. The difference $x_2 - x_1$ is defined as the velocity v, found by a given algorithm, so that applying v to x_1 gives x_2.

变式之二：离散变量

$v \leftarrow w*v + c1*rand*(pBest - p) + c2*rand*(gBest - p)$

需要重新定义 +、*、 – 等运算

重新定义的减法运算: 位置 减 位置

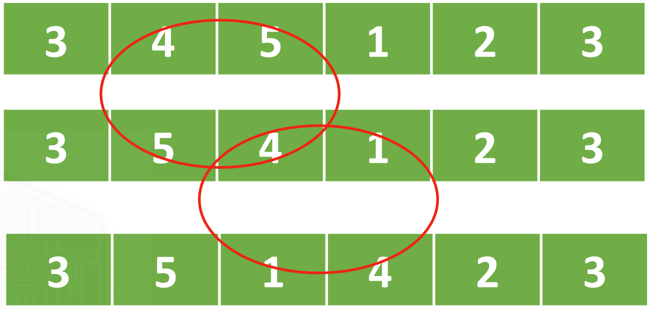


1	2	3	4	5	1

p

3	5	1	4	2	3

pBest



3	4	5	1	2	3
3	5	4	1	2	3
3	5	1	4	2	3

故pBest-p为((4, 5), (4, 1))

贝叶斯优化

Part1 如何从复杂度的角度衡量算法好坏？

Part2 针对一些有特定形式的最优化问题，如何找到最优解？

Part3 针对较复杂的最优化问题（如NP难问题），如何找到较好解？

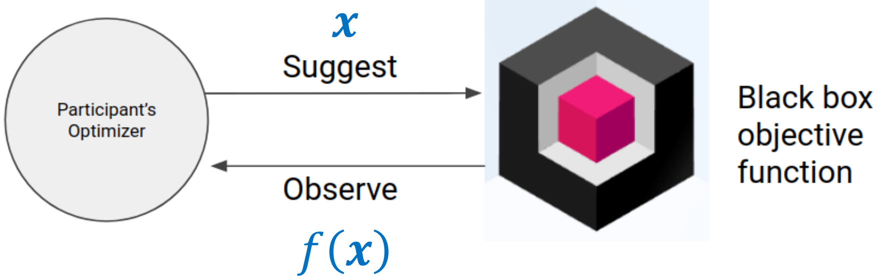
局部搜索、模拟退火、遗传算法、差分进化算法、蚁群算法、粒子群优化算法、人工蜂群算法、贝叶斯优化

Part4 针对较复杂的多目标最优化问题，如何找到较好解？

黑盒优化

自然进化策略、局部搜索、进化算法、群体智能算法等都属于黑盒优化，无需$f(\cdot)$梯度信息，仅需测试不同输入$\boldsymbol{x}$下$f(\boldsymbol{x})$的值

$$\min f(\boldsymbol{x})$$
$$\text{var } \boldsymbol{x} \in \mathcal{R}^n$$



Participant's Optimizer

$\boldsymbol{x}$

Suggest

Observe

$f(\boldsymbol{x})$

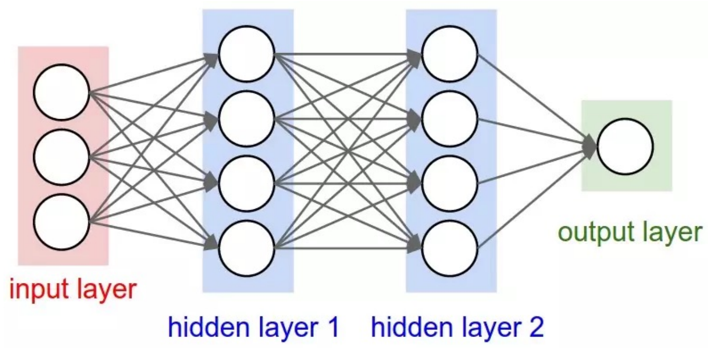
Black box objective function

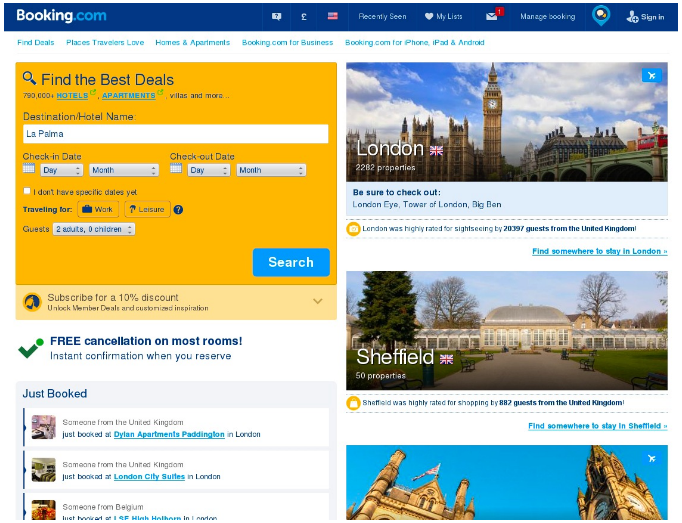
额外限制：只能测试有限个（比如几十到几百个）输入$\boldsymbol{x}$下$f(\boldsymbol{x})$的值

黑盒优化

获得$f(x)$的过程可能耗时长、开销大：

- ❑ 神经网络的超参数（学习率、丢弃率等）优化
- ❑ 大规模人群实验的优化
- ❑ 大规模计算仿真/物理/化学等实验的优化





Find the Best Deals

790,000+ HOTELS, APARTMENTS, villas and more...

Destination/Hotel Name: La Palma

Check-in Date: Day Month

Check-out Date: Day Month

☐ I don't have specific dates yet

Traveling for: Work Leisure

Guests: 2 adults, 0 children

Subscribe for a 10% discount
Unlock Member Deals and customized inspiration

FREE cancellation on most rooms!
Instant confirmation when you reserve

Just Booked

Someone from the United Kingdom
just booked at Dylan Apartments Paddington in London

Someone from the United Kingdom
just booked at London City Suites in London

Someone from Belgium
just booked at LSE High Holborn in London

London
2282 properties

Be sure to check out:
London Eye, Tower of London, Big Ben

London was highly rated for sightseeing by **20397 guests from the United Kingdom**!

Find somewhere to stay in London »

Sheffield
50 properties

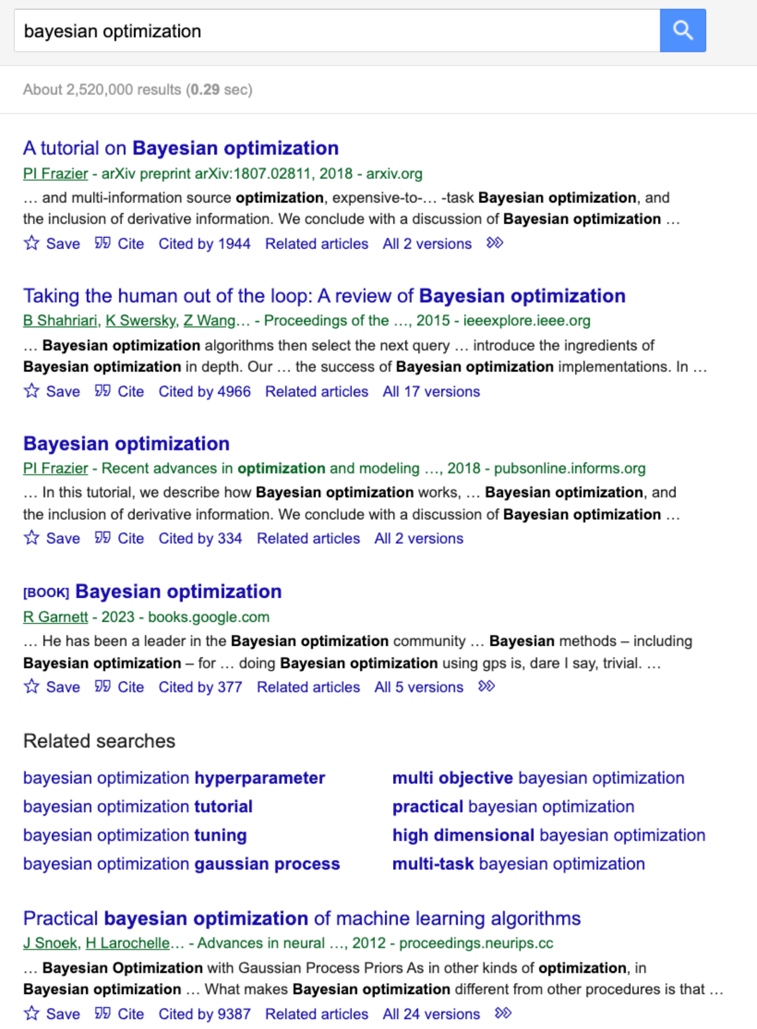
Sheffield was highly rated for shopping by **882 guests from the United Kingdom**!

Find somewhere to stay in Sheffield »

贝叶斯优化

贝叶斯优化是目前最常用的求解此类问题的方法，一般对$n \leq 20$的情况尤其适用

$$\min f(\boldsymbol{x})$$
$$\text{var } \boldsymbol{x} \in \mathcal{R}^n$$



bayesian optimization

About 2,520,000 results (**0.29** sec)

A tutorial on **Bayesian optimization**
PI Frazier - arXiv preprint arXiv:1807.02811, 2018 - arxiv.org
… and multi-information source **optimization**, expensive-to-… -task **Bayesian optimization**, and the inclusion of derivative information. We conclude with a discussion of **Bayesian optimization** …
☆ Save 99 Cite Cited by 1944 Related articles All 2 versions »

Taking the human out of the loop: A review of **Bayesian optimization**
B Shahriari, K Swersky, Z Wang… - Proceedings of the …, 2015 - ieeexplore.ieee.org
… **Bayesian optimization** algorithms then select the next query … introduce the ingredients of **Bayesian optimization** in depth. Our … the success of **Bayesian optimization** implementations. In …
☆ Save 99 Cite Cited by 4966 Related articles All 17 versions

Bayesian optimization
PI Frazier - Recent advances in **optimization** and modeling …, 2018 - pubsonline.informs.org
… In this tutorial, we describe how **Bayesian optimization** works, … **Bayesian optimization**, and the inclusion of derivative information. We conclude with a discussion of **Bayesian optimization** …
☆ Save 99 Cite Cited by 334 Related articles All 2 versions

[BOOK] **Bayesian optimization**
R Garnett - 2023 - books.google.com
… He has been a leader in the **Bayesian optimization** community … **Bayesian** methods – including **Bayesian optimization** – for … doing **Bayesian optimization** using gps is, dare I say, trivial. …
☆ Save 99 Cite Cited by 377 Related articles All 5 versions »

Related searches

bayesian optimization **hyperparameter**	**multi objective** bayesian optimization
bayesian optimization **tutorial**	**practical** bayesian optimization
bayesian optimization **tuning**	**high dimensional** bayesian optimization
bayesian optimization **gaussian process**	**multi-task** bayesian optimization

Practical **bayesian optimization** of machine learning algorithms
J Snoek, H Larochelle… - Advances in neural …, 2012 - proceedings.neurips.cc
… **Bayesian Optimization** with Gaussian Process Priors As in other kinds of **optimization**, in **Bayesian optimization** … What makes **Bayesian optimization** different from other procedures is that …
☆ Save 99 Cite Cited by 9387 Related articles All 24 versions »

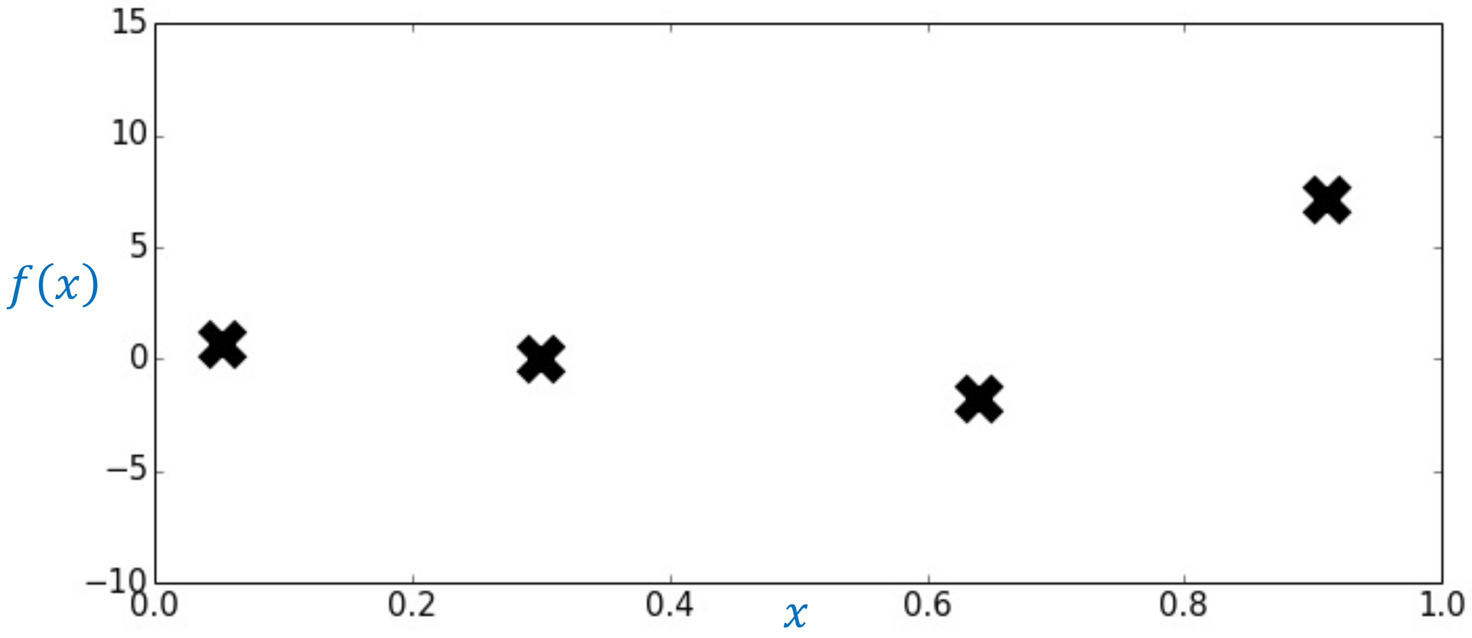
贝叶斯优化

贝叶斯优化是目前最常用的求解此类问题的方法，一般对$n \leq 20$的情况尤其适用

贝叶斯定理

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

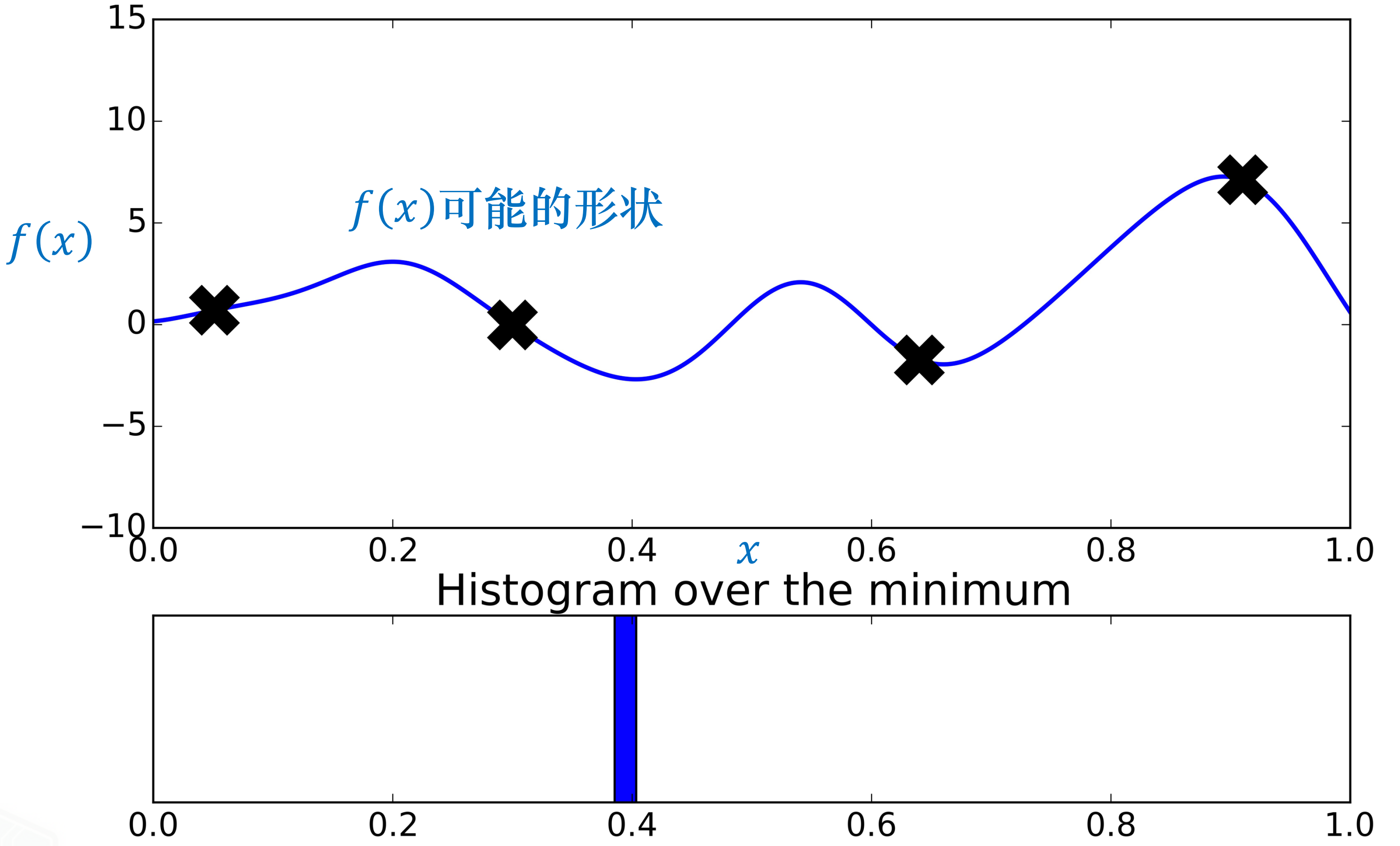
贝叶斯优化



考虑变量是一维的例子，已知四个不同x的$f(x)$值，猜测最优解x^*在哪里?

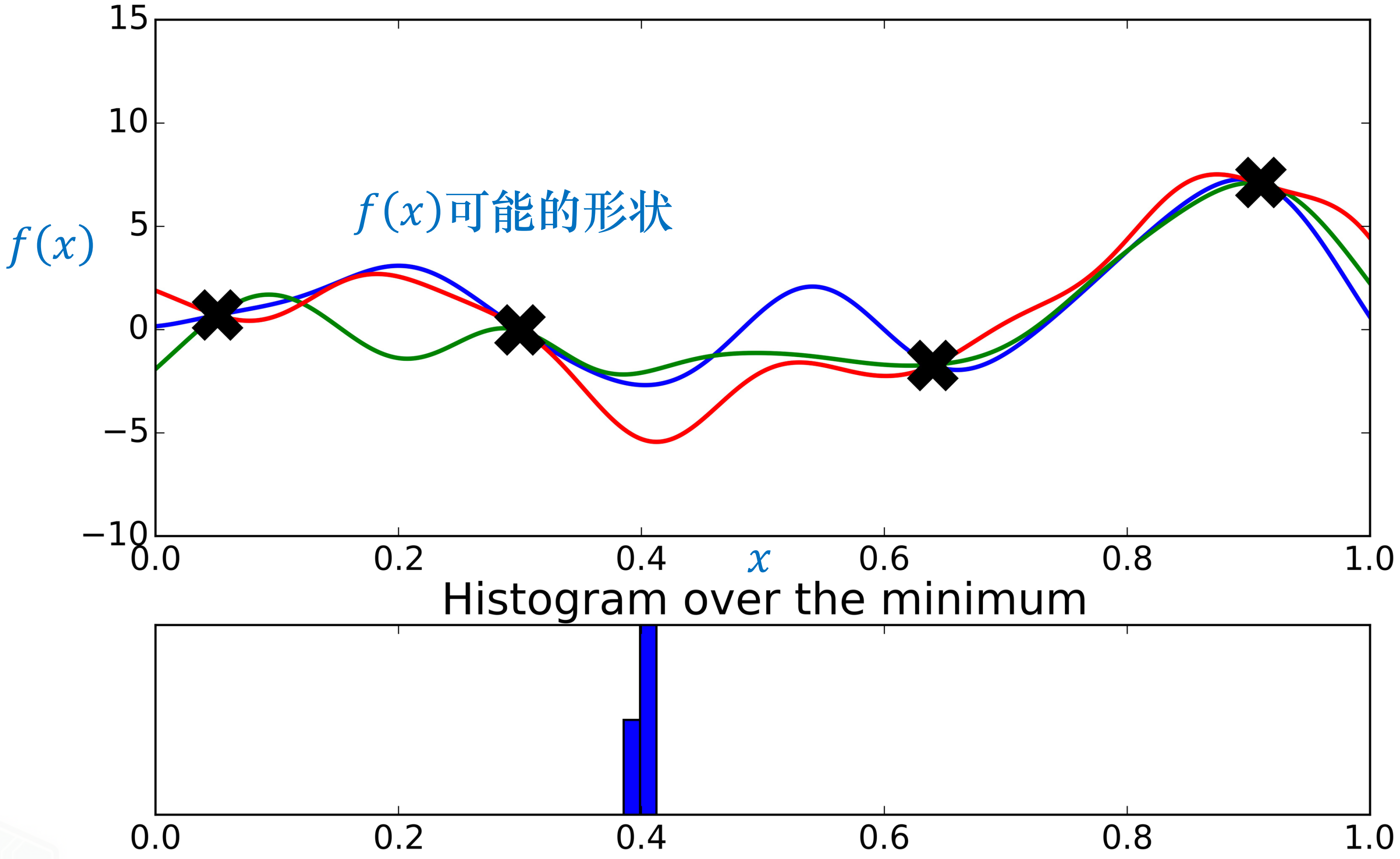
图示取自Javier Gonzalez <Introduction to Bayesian Optimization> Slides

贝叶斯优化



在该$f(x)$形状下，$x^* = 0.4$

贝叶斯优化

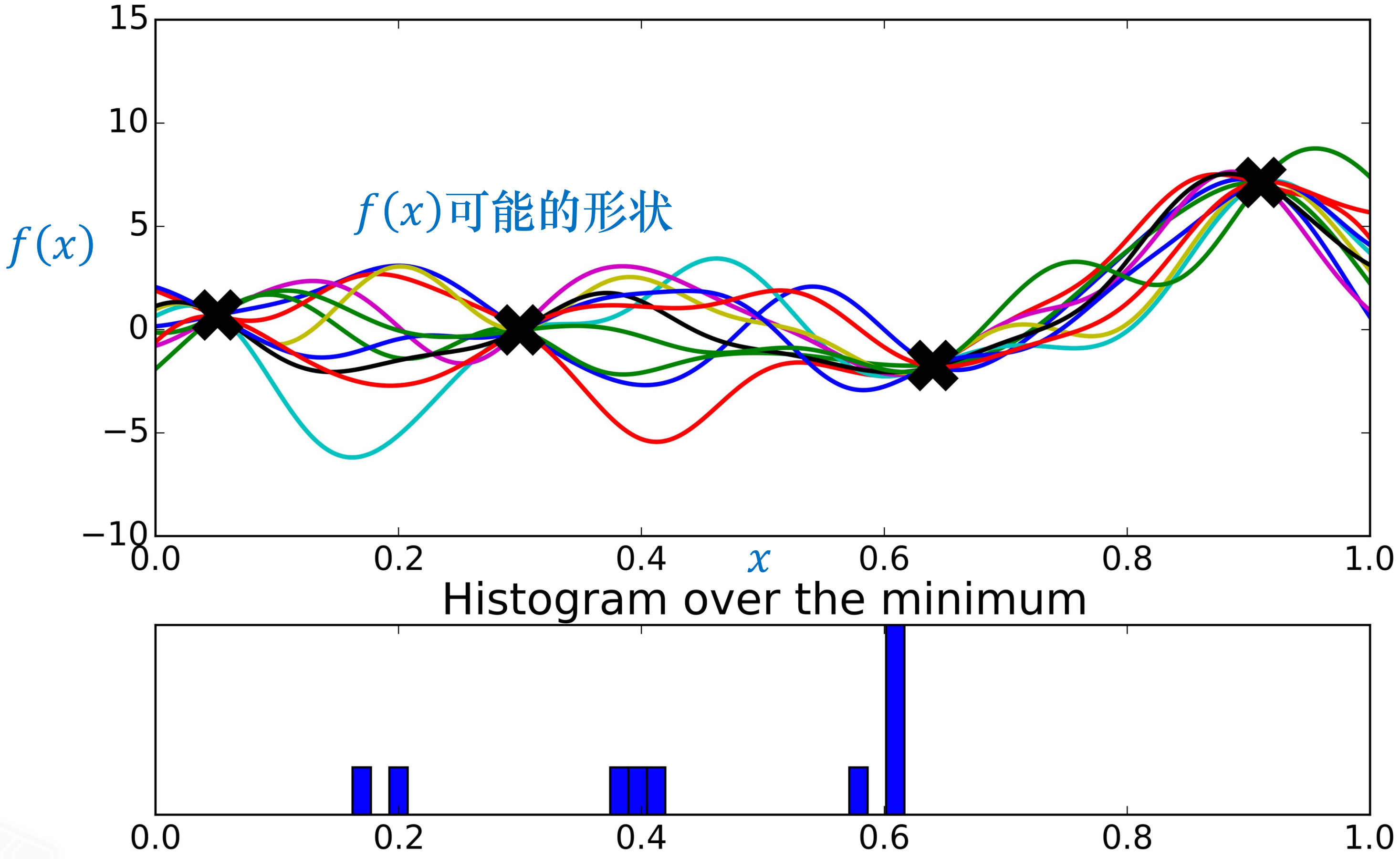


$f(x)$可能的形状

Histogram over the minimum

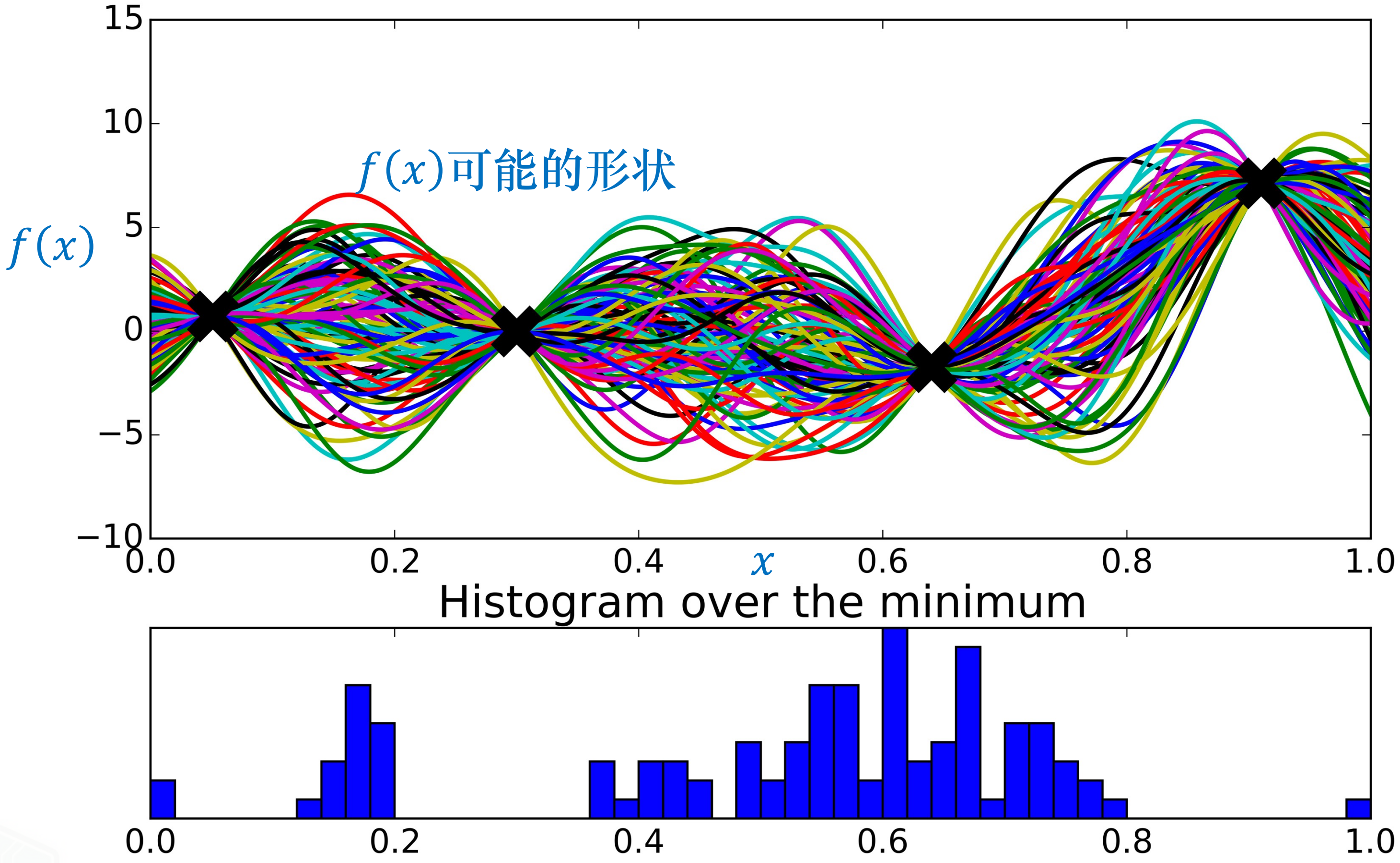
在以上三种$f(x)$形状下， x^*分别取0.39,0.4,0.4

贝叶斯优化



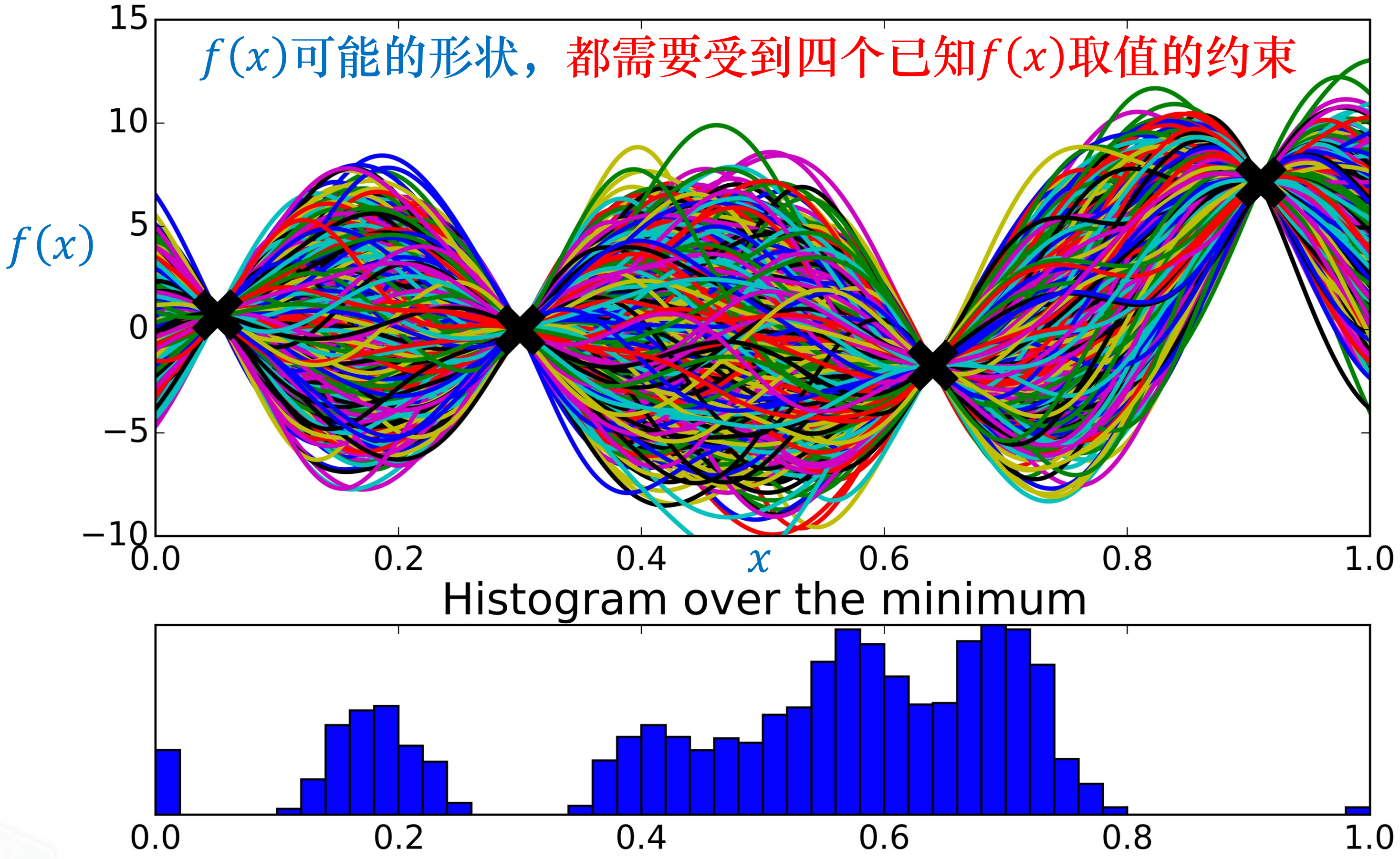
下图显示x^*的频率分布

贝叶斯优化



下图显示x^*的频率分布

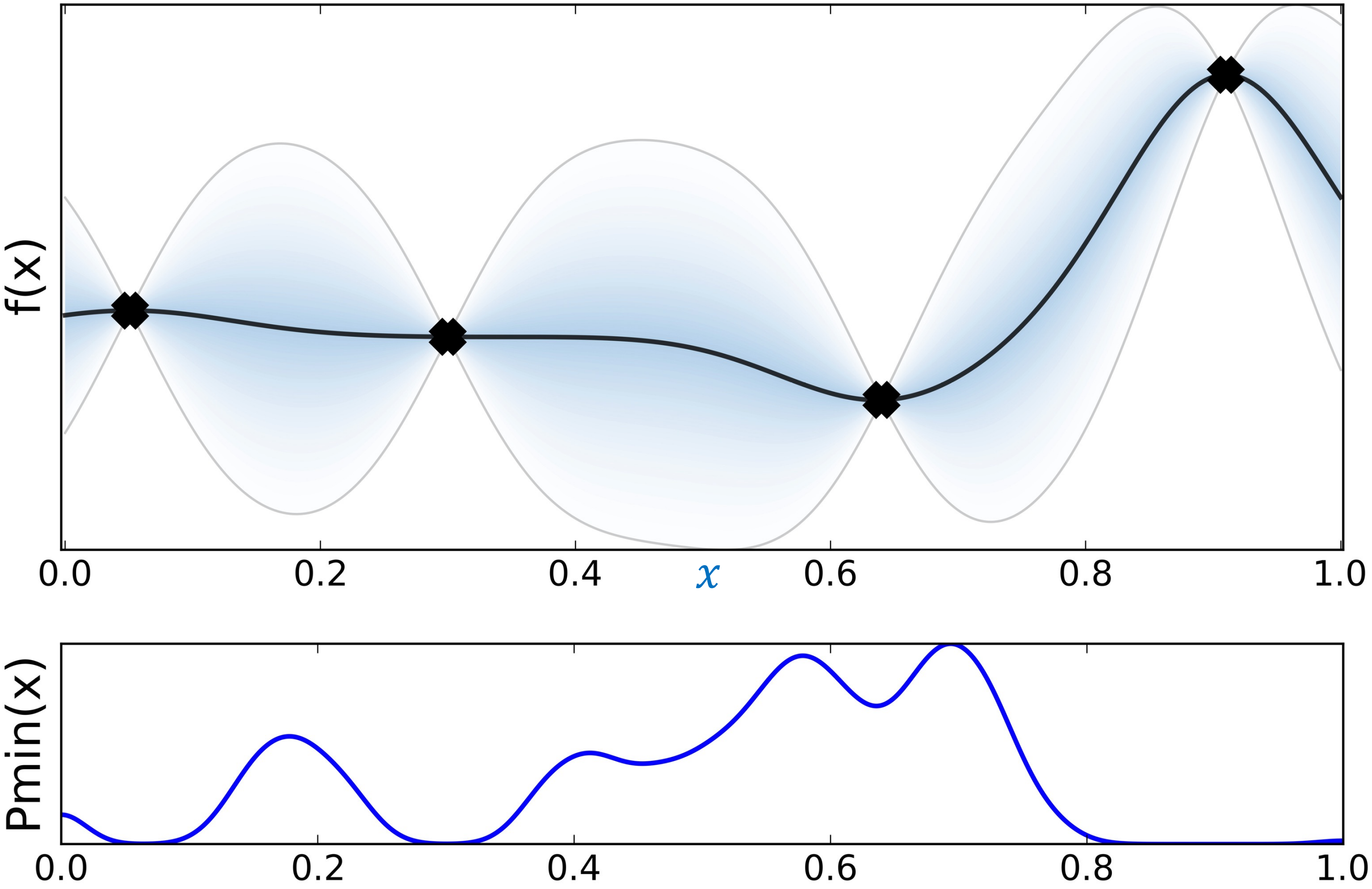
贝叶斯优化



$f(x)$可能的形状，都需要受到四个已知$f(x)$取值的约束

下图显示x^*的频率分布

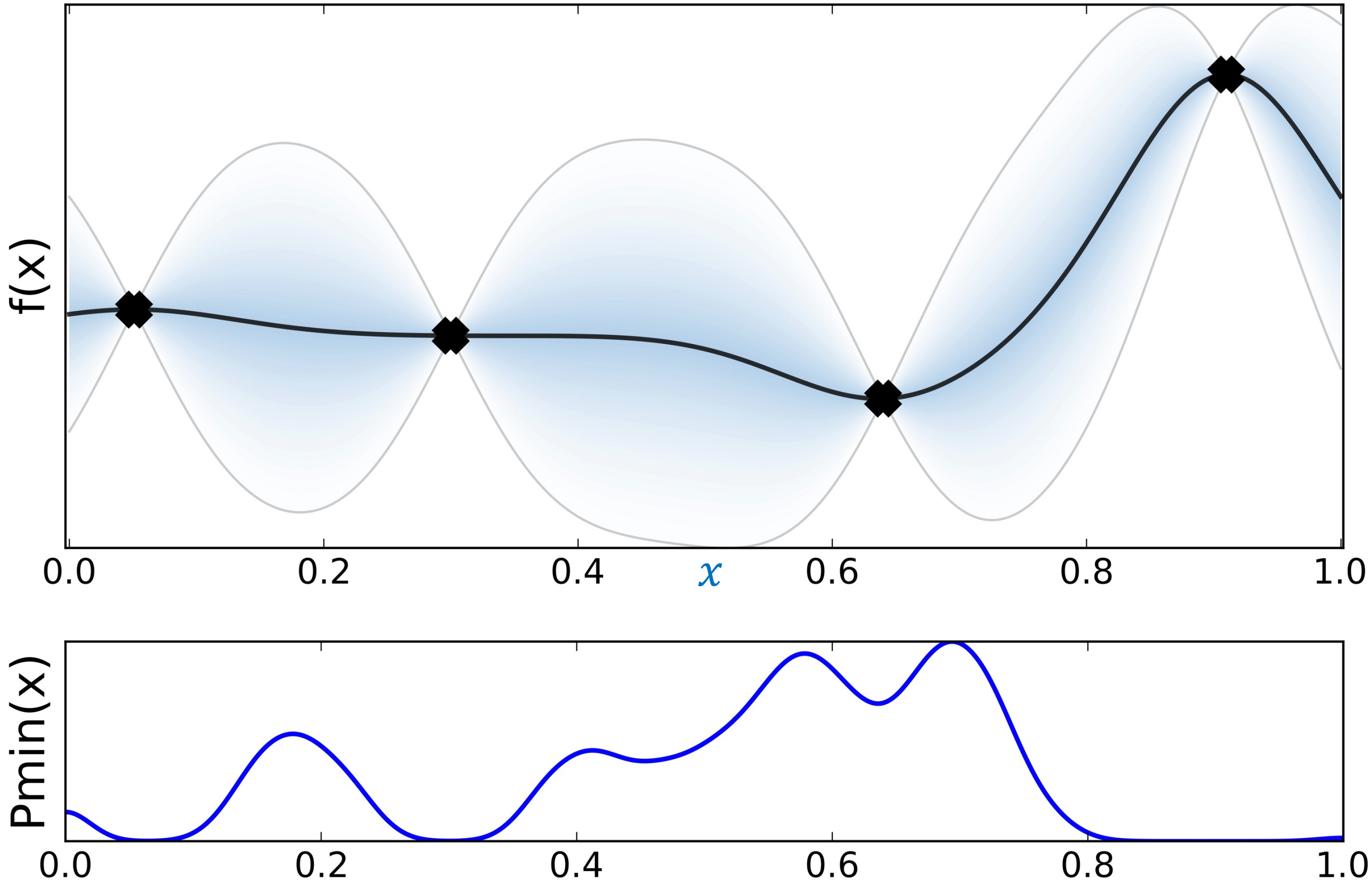
贝叶斯优化



$f(x)$有无数种可能的形状。如果考虑完$f(x)$的所有可能，可以得到x^*的概率分布

贝叶斯优化

如果可以刻画在给定四个 x 下 $f(x)$ 函数形状的分布，就能计算此时x^*的概率分布



$f(x)$有无数种可能的形状。如果考虑完$f(x)$的所有可能，可以得到x^*的概率分布

高斯过程

如何刻画$f(\boldsymbol{x})$函数形状的分布？其中$\boldsymbol{x} \in \mathcal{R}^n$

假设，对任意K个解$\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_K$，函数相应的$K$个取值服从$K$维高斯分布：

$$\begin{bmatrix} f(\boldsymbol{x}_1) \\ f(\boldsymbol{x}_2) \\ \cdots \\ f(\boldsymbol{x}_K) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu(\boldsymbol{x}_1) \\ \mu(\boldsymbol{x}_2) \\ \cdots \\ \mu(\boldsymbol{x}_K) \end{bmatrix}, \begin{bmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}_1) & k(\boldsymbol{x}_1, \boldsymbol{x}_2) & \dots & k(\boldsymbol{x}_1, \boldsymbol{x}_K) \\ k(\boldsymbol{x}_2, \boldsymbol{x}_1) & k(\boldsymbol{x}_2, \boldsymbol{x}_2) & \dots & k(\boldsymbol{x}_2, \boldsymbol{x}_K) \\ \cdots & \cdots & \cdots & \cdots \\ k(\boldsymbol{x}_K, \boldsymbol{x}_1) & k(\boldsymbol{x}_K, \boldsymbol{x}_2) & \dots & k(\boldsymbol{x}_K, \boldsymbol{x}_K) \end{bmatrix} \right)$$

K×1列向量　　K×1列向量　　K×K矩阵

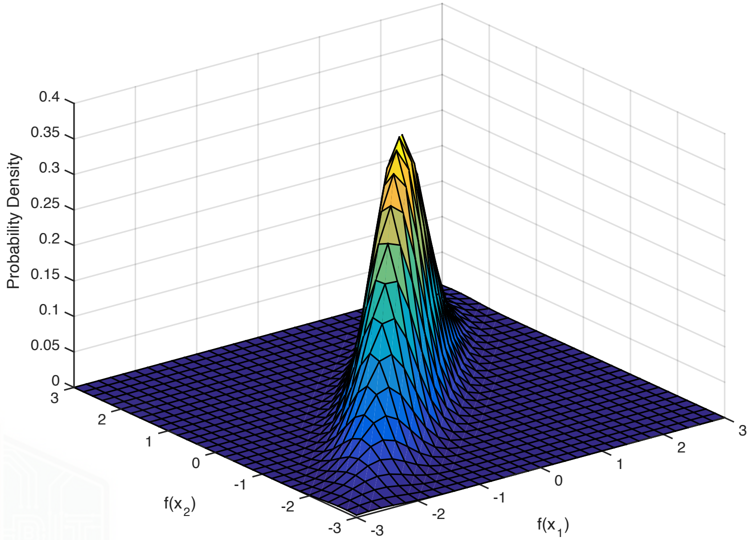
其中，$\mu(\cdot)$和$k(\cdot,\cdot)$是假设指定的函数

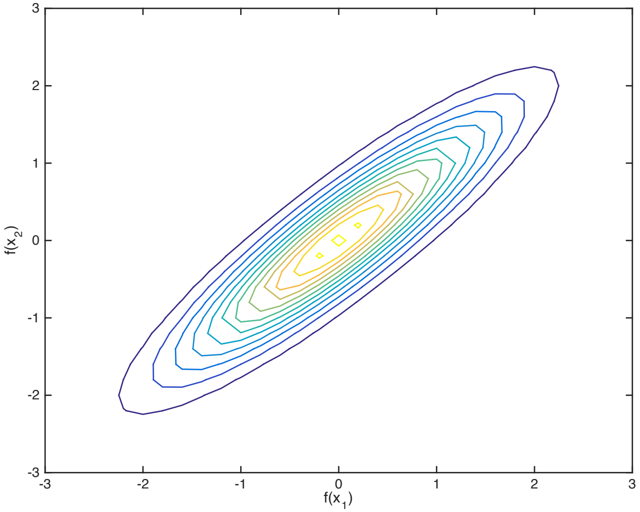
高斯过程

例如：$K=2, \mu(\boldsymbol{x})=0, k(\boldsymbol{x}_1,\boldsymbol{x}_2)=\exp(-\|\boldsymbol{x}_1-\boldsymbol{x}_2\|_2^2)$

$$\begin{bmatrix} f(\boldsymbol{x}_1) \\ f(\boldsymbol{x}_2) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & \exp(-\|\boldsymbol{x}_1-\boldsymbol{x}_2\|_2^2) \\ \exp(-\|\boldsymbol{x}_1-\boldsymbol{x}_2\|_2^2) & 1 \end{bmatrix}\right)$$

若$\|\boldsymbol{x}_1-\boldsymbol{x}_2\|_2^2=0.1$，$\boldsymbol{x}_1$与$\boldsymbol{x}_2$距离近，$f(\boldsymbol{x}_1)$与$f(\boldsymbol{x}_2)$正相关性强



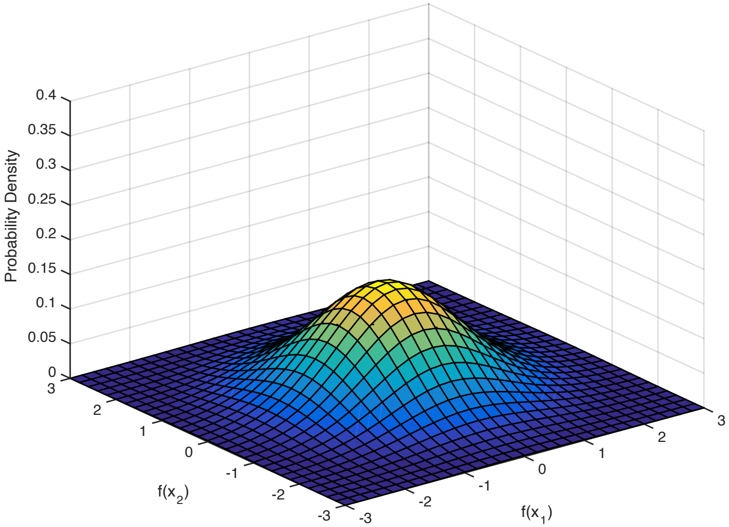


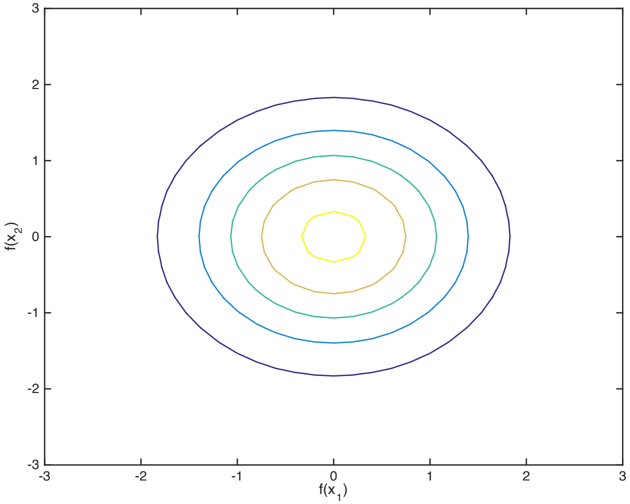
高斯过程

例如：$K=2, \mu(\boldsymbol{x})=0, k(\boldsymbol{x}_1,\boldsymbol{x}_2)=\exp(-\|\boldsymbol{x}_1-\boldsymbol{x}_2\|_2^2)$

$$\begin{bmatrix} f(\boldsymbol{x}_1) \\ f(\boldsymbol{x}_2) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & \exp(-\|\boldsymbol{x}_1-\boldsymbol{x}_2\|_2^2) \\ \exp(-\|\boldsymbol{x}_1-\boldsymbol{x}_2\|_2^2) & 1 \end{bmatrix}\right)$$

若$\|\boldsymbol{x}_1-\boldsymbol{x}_2\|_2^2=10$，$\boldsymbol{x}_1$与$\boldsymbol{x}_2$距离远，$f(\boldsymbol{x}_1)$与$f(\boldsymbol{x}_2)$相关性弱





高斯过程

如何刻画$f(\boldsymbol{x})$函数形状的分布？ 其中$\boldsymbol{x} \in \mathcal{R}^n$

假设，对任意K个解$\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_K$，函数相应的$K$个取值服从$K$维高斯分布：

$$\begin{bmatrix} f(\boldsymbol{x}_1) \\ f(\boldsymbol{x}_2) \\ \cdots \\ f(\boldsymbol{x}_K) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu(\boldsymbol{x}_1) \\ \mu(\boldsymbol{x}_2) \\ \cdots \\ \mu(\boldsymbol{x}_K) \end{bmatrix}, \begin{bmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}_1) & k(\boldsymbol{x}_1, \boldsymbol{x}_2) & \ldots & k(\boldsymbol{x}_1, \boldsymbol{x}_K) \\ k(\boldsymbol{x}_2, \boldsymbol{x}_1) & k(\boldsymbol{x}_2, \boldsymbol{x}_2) & \ldots & k(\boldsymbol{x}_2, \boldsymbol{x}_K) \\ \cdots & \cdots & \cdots & \cdots \\ k(\boldsymbol{x}_K, \boldsymbol{x}_1) & k(\boldsymbol{x}_K, \boldsymbol{x}_2) & \ldots & k(\boldsymbol{x}_K, \boldsymbol{x}_K) \end{bmatrix} \right)$$

K×1列向量　　K×1列向量　　K×K矩阵

其中，$\mu(\cdot)$和$k(\cdot,\cdot)$是假设指定的函数

若已知四个不同$\boldsymbol{x}$的$f(\boldsymbol{x})$值，如何利用以上假设分析最优解$\boldsymbol{x}^*$的位置？

高斯过程

根据假设，已知$f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), f(\boldsymbol{x}_3), f(\boldsymbol{x}_4)$，那么对任意$f(\boldsymbol{x})$有：

$f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), f(\boldsymbol{x}_3), f(\boldsymbol{x}_4), f(\boldsymbol{x})$服从5维高斯分布：

$$\begin{bmatrix} f(\boldsymbol{x}_1) \\ f(\boldsymbol{x}_2) \\ f(\boldsymbol{x}_3) \\ f(\boldsymbol{x}_4) \\ f(\boldsymbol{x}) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu(\boldsymbol{x}_1) \\ \mu(\boldsymbol{x}_2) \\ \mu(\boldsymbol{x}_3) \\ \mu(\boldsymbol{x}_4) \\ \mu(\boldsymbol{x}) \end{bmatrix}, \begin{bmatrix} k(\boldsymbol{x}_1,\boldsymbol{x}_1) & k(\boldsymbol{x}_1,\boldsymbol{x}_2) & k(\boldsymbol{x}_1,\boldsymbol{x}_3) & k(\boldsymbol{x}_1,\boldsymbol{x}_4) & k(\boldsymbol{x}_1,\boldsymbol{x}) \\ k(\boldsymbol{x}_2,\boldsymbol{x}_1) & k(\boldsymbol{x}_2,\boldsymbol{x}_2) & k(\boldsymbol{x}_2,\boldsymbol{x}_3) & k(\boldsymbol{x}_2,\boldsymbol{x}_4) & k(\boldsymbol{x}_2,\boldsymbol{x}) \\ k(\boldsymbol{x}_3,\boldsymbol{x}_1) & k(\boldsymbol{x}_3,\boldsymbol{x}_2) & k(\boldsymbol{x}_3,\boldsymbol{x}_3) & k(\boldsymbol{x}_3,\boldsymbol{x}_4) & k(\boldsymbol{x}_3,\boldsymbol{x}) \\ k(\boldsymbol{x}_4,\boldsymbol{x}_1) & k(\boldsymbol{x}_4,\boldsymbol{x}_2) & k(\boldsymbol{x}_4,\boldsymbol{x}_3) & k(\boldsymbol{x}_4,\boldsymbol{x}_4) & k(\boldsymbol{x}_4,\boldsymbol{x}) \\ k(\boldsymbol{x},\boldsymbol{x}_1) & k(\boldsymbol{x},\boldsymbol{x}_2) & k(\boldsymbol{x},\boldsymbol{x}_3) & k(\boldsymbol{x},\boldsymbol{x}_4) & k(\boldsymbol{x},\boldsymbol{x}) \end{bmatrix} \right)$$

5×1列向量　　5×1列向量　　5×5矩阵

高斯过程

根据假设，已知$f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), f(\boldsymbol{x}_3), f(\boldsymbol{x}_4)$，那么对任意$f(\boldsymbol{x})$有:

$f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), f(\boldsymbol{x}_3), f(\boldsymbol{x}_4), f(\boldsymbol{x})$服从5维高斯分布:

$$\begin{bmatrix} f(\boldsymbol{x}_1) \\ f(\boldsymbol{x}_2) \\ f(\boldsymbol{x}_3) \\ f(\boldsymbol{x}_4) \\ f(\boldsymbol{x}) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu(\boldsymbol{x}_1) \\ \mu(\boldsymbol{x}_2) \\ \mu(\boldsymbol{x}_3) \\ \mu(\boldsymbol{x}_4) \\ \mu(\boldsymbol{x}) \end{bmatrix}, \begin{bmatrix} k(\boldsymbol{x}_1,\boldsymbol{x}_1) & k(\boldsymbol{x}_1,\boldsymbol{x}_2) & k(\boldsymbol{x}_1,\boldsymbol{x}_3) & k(\boldsymbol{x}_1,\boldsymbol{x}_4) & k(\boldsymbol{x}_1,\boldsymbol{x}) \\ k(\boldsymbol{x}_2,\boldsymbol{x}_1) & k(\boldsymbol{x}_2,\boldsymbol{x}_2) & k(\boldsymbol{x}_2,\boldsymbol{x}_3) & k(\boldsymbol{x}_2,\boldsymbol{x}_4) & k(\boldsymbol{x}_2,\boldsymbol{x}) \\ k(\boldsymbol{x}_3,\boldsymbol{x}_1) & k(\boldsymbol{x}_3,\boldsymbol{x}_2) & k(\boldsymbol{x}_3,\boldsymbol{x}_3) & k(\boldsymbol{x}_3,\boldsymbol{x}_4) & k(\boldsymbol{x}_3,\boldsymbol{x}) \\ k(\boldsymbol{x}_4,\boldsymbol{x}_1) & k(\boldsymbol{x}_4,\boldsymbol{x}_2) & k(\boldsymbol{x}_4,\boldsymbol{x}_3) & k(\boldsymbol{x}_4,\boldsymbol{x}_4) & k(\boldsymbol{x}_4,\boldsymbol{x}) \\ k(\boldsymbol{x},\boldsymbol{x}_1) & k(\boldsymbol{x},\boldsymbol{x}_2) & k(\boldsymbol{x},\boldsymbol{x}_3) & k(\boldsymbol{x},\boldsymbol{x}_4) & k(\boldsymbol{x},\boldsymbol{x}) \end{bmatrix} \right)$$

5×1列向量　　5×1列向量　　5×5矩阵

利用贝叶斯定理 $P(A|B) = \dfrac{P(B|A)P(A)}{P(B)}$ 可以分析$f(\boldsymbol{x})$的概率分布

高斯过程

根据假设，已知$f(\boldsymbol{x}_1),f(\boldsymbol{x}_2),f(\boldsymbol{x}_3),f(\boldsymbol{x}_4)$，那么对任意$f(\boldsymbol{x})$有：

$f(\boldsymbol{x}_1),f(\boldsymbol{x}_2),f(\boldsymbol{x}_3),f(\boldsymbol{x}_4),f(\boldsymbol{x})$服从5维高斯分布：

$$\begin{bmatrix} f(\boldsymbol{x}_1) \\ f(\boldsymbol{x}_2) \\ f(\boldsymbol{x}_3) \\ f(\boldsymbol{x}_4) \\ f(\boldsymbol{x}) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu(\boldsymbol{x}_1) \\ \mu(\boldsymbol{x}_2) \\ \mu(\boldsymbol{x}_3) \\ \mu(\boldsymbol{x}_4) \\ \mu(\boldsymbol{x}) \end{bmatrix}, \begin{bmatrix} k(\boldsymbol{x}_1,\boldsymbol{x}_1) & k(\boldsymbol{x}_1,\boldsymbol{x}_2) & k(\boldsymbol{x}_1,\boldsymbol{x}_3) & k(\boldsymbol{x}_1,\boldsymbol{x}_4) & k(\boldsymbol{x}_1,\boldsymbol{x}) \\ k(\boldsymbol{x}_2,\boldsymbol{x}_1) & k(\boldsymbol{x}_2,\boldsymbol{x}_2) & k(\boldsymbol{x}_2,\boldsymbol{x}_3) & k(\boldsymbol{x}_2,\boldsymbol{x}_4) & k(\boldsymbol{x}_2,\boldsymbol{x}) \\ k(\boldsymbol{x}_3,\boldsymbol{x}_1) & k(\boldsymbol{x}_3,\boldsymbol{x}_2) & k(\boldsymbol{x}_3,\boldsymbol{x}_3) & k(\boldsymbol{x}_3,\boldsymbol{x}_4) & k(\boldsymbol{x}_3,\boldsymbol{x}) \\ k(\boldsymbol{x}_4,\boldsymbol{x}_1) & k(\boldsymbol{x}_4,\boldsymbol{x}_2) & k(\boldsymbol{x}_4,\boldsymbol{x}_3) & k(\boldsymbol{x}_4,\boldsymbol{x}_4) & k(\boldsymbol{x}_4,\boldsymbol{x}) \\ k(\boldsymbol{x},\boldsymbol{x}_1) & k(\boldsymbol{x},\boldsymbol{x}_2) & k(\boldsymbol{x},\boldsymbol{x}_3) & k(\boldsymbol{x},\boldsymbol{x}_4) & k(\boldsymbol{x},\boldsymbol{x}) \end{bmatrix} \right)$$

5×1列向量　　5×1列向量　　5×5矩阵

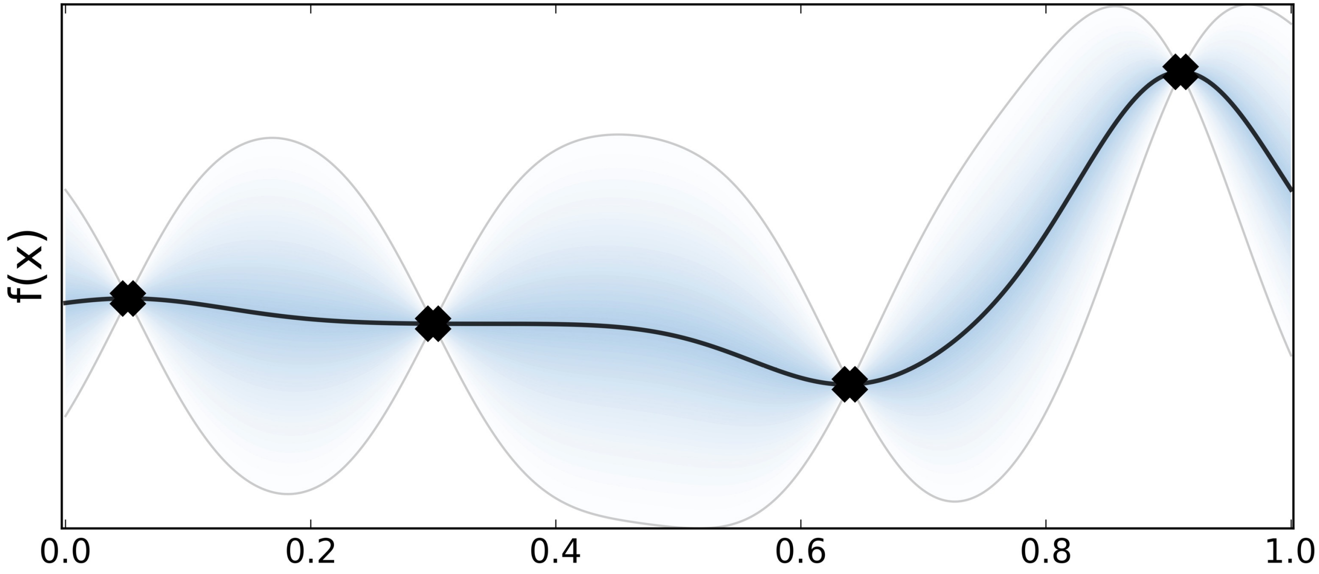
$$\Pr\big(f(\boldsymbol{x})|f(\boldsymbol{x}_1),f(\boldsymbol{x}_2),f(\boldsymbol{x}_3),f(\boldsymbol{x}_4)\big) = \frac{\Pr\big(f(\boldsymbol{x}),f(\boldsymbol{x}_1),f(\boldsymbol{x}_2),f(\boldsymbol{x}_3),f(\boldsymbol{x}_4)\big)}{\Pr\big(f(\boldsymbol{x}_1),f(\boldsymbol{x}_2),f(\boldsymbol{x}_3),f(\boldsymbol{x}_4)\big)}$$

结论：此时$f(\boldsymbol{x})$服从1维高斯分布，均值记为$\tilde{\mu}(\boldsymbol{x})$、标准差记为$\tilde{\sigma}(\boldsymbol{x})$（具体形式推导略）

贝叶斯优化

已知$f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), f(\boldsymbol{x}_3), f(\boldsymbol{x}_4)$，那么此时$f(\boldsymbol{x})$服从1维高斯分布

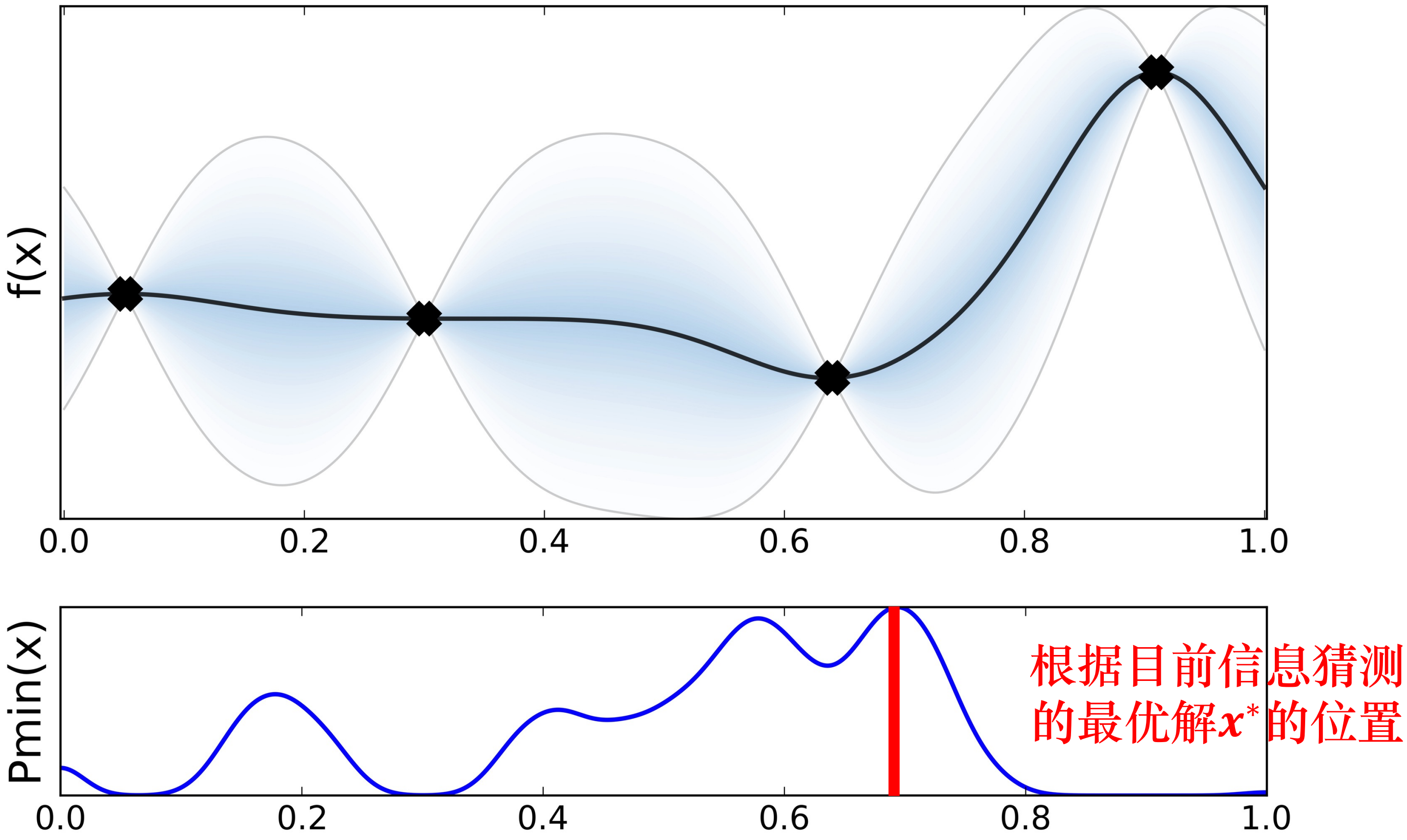
（均值$\tilde{\mu}(\boldsymbol{x})$和标准差$\tilde{\sigma}(\boldsymbol{x})$是根据$f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), f(\boldsymbol{x}_3), f(\boldsymbol{x}_4)$计算而得）



贝叶斯优化

已知$f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), f(\boldsymbol{x}_3), f(\boldsymbol{x}_4)$，那么此时$f(\boldsymbol{x})$服从1维高斯分布

(均值$\tilde{\mu}(\boldsymbol{x})$和标准差$\tilde{\sigma}(\boldsymbol{x})$是根据$f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), f(\boldsymbol{x}_3), f(\boldsymbol{x}_4)$计算而得)

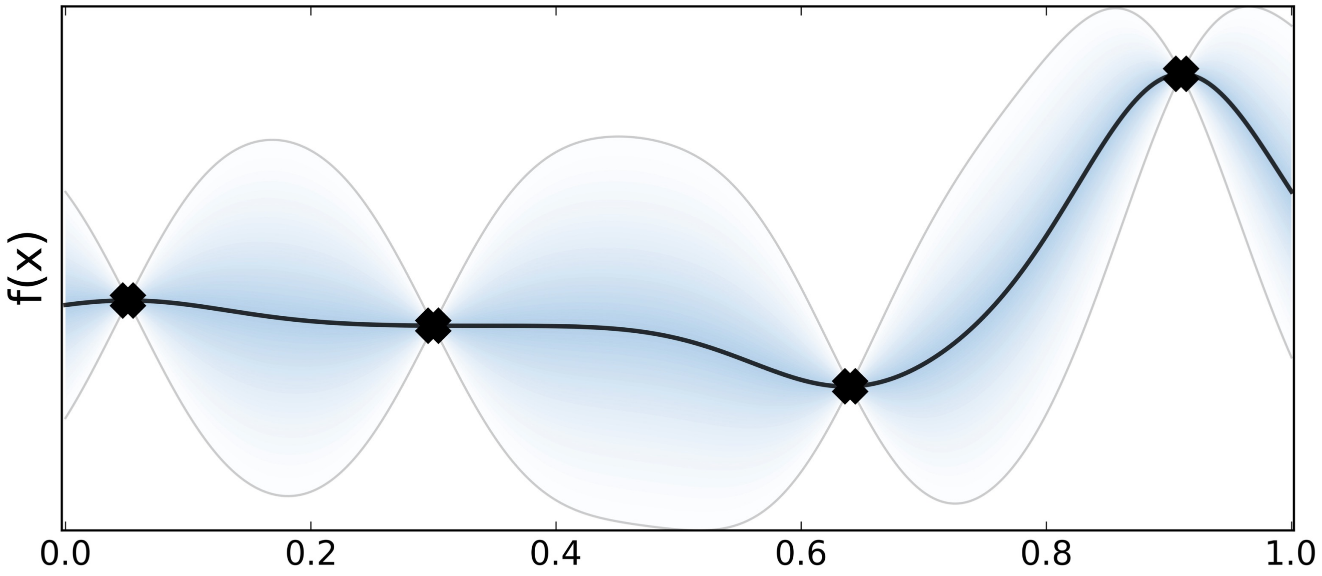


根据目前信息猜测的最优解$\boldsymbol{x}^*$的位置

贝叶斯优化

已知$f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), f(\boldsymbol{x}_3), f(\boldsymbol{x}_4)$，那么此时$f(\boldsymbol{x})$服从1维高斯分布

（均值$\tilde{\mu}(\boldsymbol{x})$和标准差$\tilde{\sigma}(\boldsymbol{x})$是根据$f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), f(\boldsymbol{x}_3), f(\boldsymbol{x}_4)$计算而得）



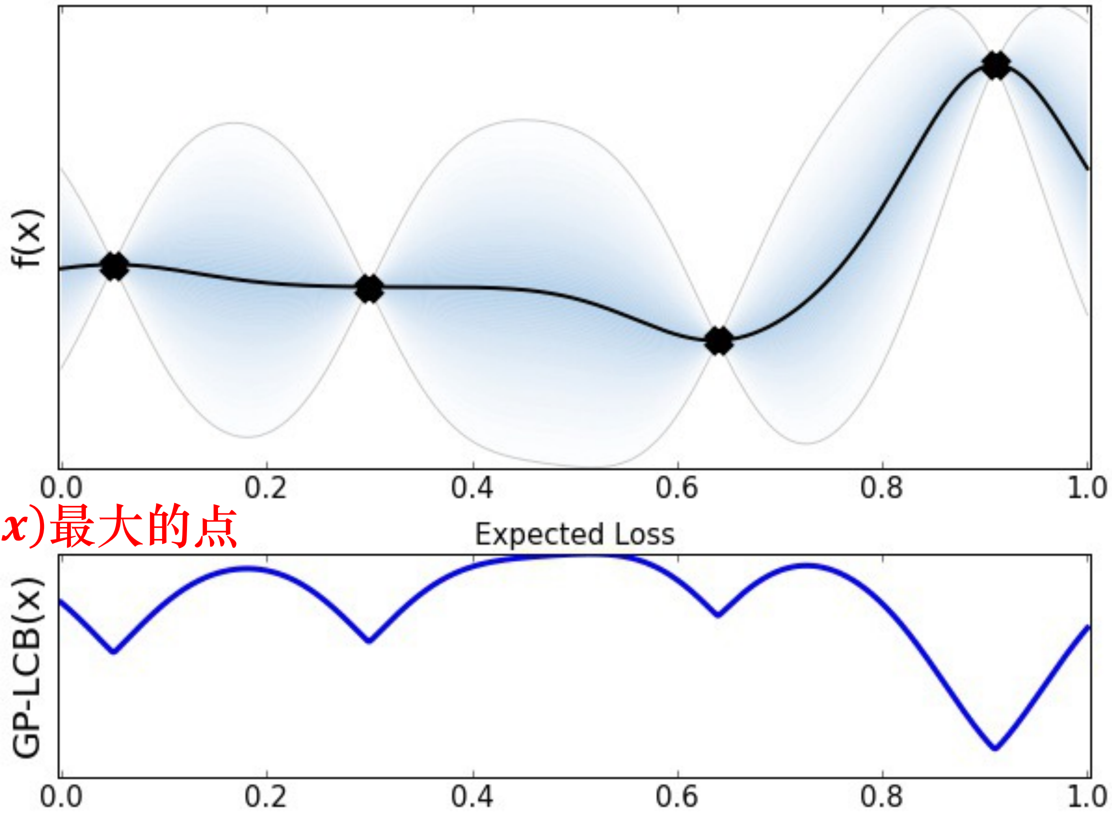
如果共有5次测试机会，下一步应该选哪个点测试?

应该尽量获得更多关于最优解$\boldsymbol{x}^*$的信息

贝叶斯优化

已知$f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), f(\boldsymbol{x}_3), f(\boldsymbol{x}_4)$，那么此时$f(\boldsymbol{x})$服从1维高斯分布

（均值$\tilde{\mu}(\boldsymbol{x})$和标准差$\tilde{\sigma}(\boldsymbol{x})$是根据$f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), f(\boldsymbol{x}_3), f(\boldsymbol{x}_4)$计算而得）



选择令$-\tilde{\mu}(\boldsymbol{x}) + \tilde{\sigma}(\boldsymbol{x})$最大的点

兼顾exploitation

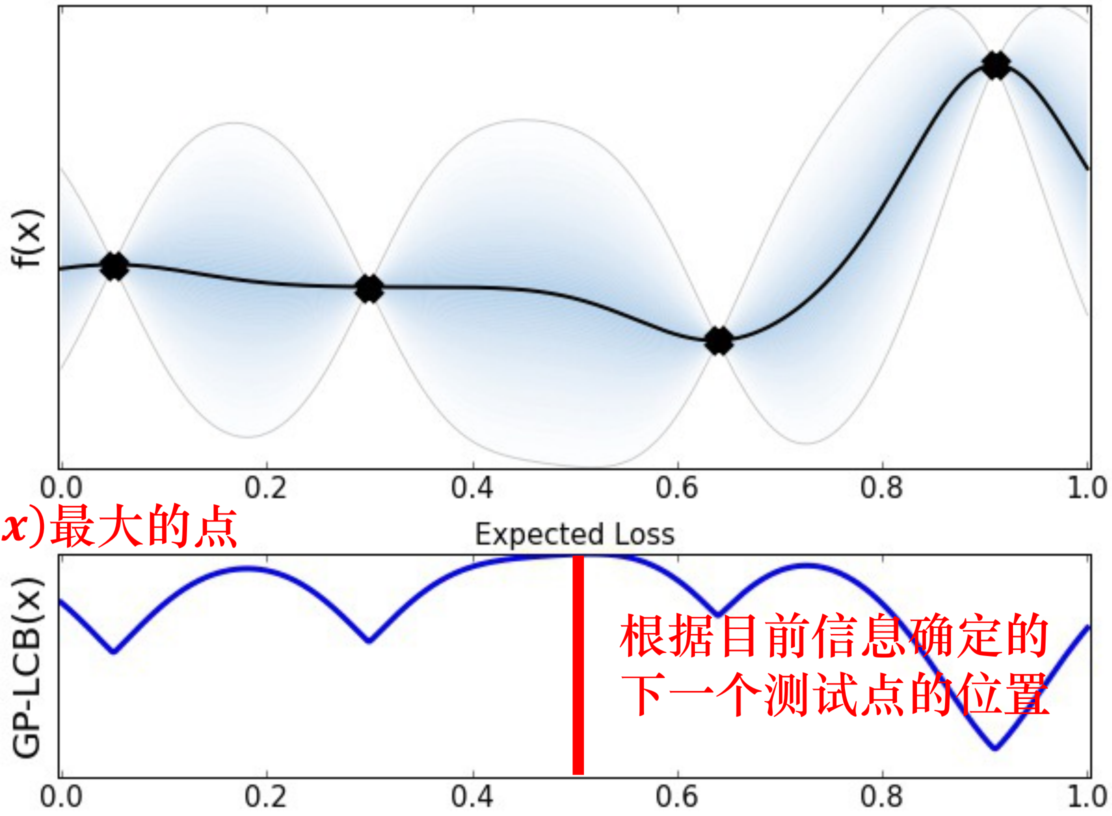
与exploration

（标准不唯一）

贝叶斯优化

已知$f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), f(\boldsymbol{x}_3), f(\boldsymbol{x}_4)$，那么此时$f(\boldsymbol{x})$服从1维高斯分布

（均值$\tilde{\mu}(\boldsymbol{x})$和标准差$\tilde{\sigma}(\boldsymbol{x})$是根据$f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), f(\boldsymbol{x}_3), f(\boldsymbol{x}_4)$计算而得）



根据目前信息确定的下一个测试点的位置

选择令$-\tilde{\mu}(\boldsymbol{x})+\tilde{\sigma}(\boldsymbol{x})$最大的点

兼顾exploitation

与exploration

（标准不唯一）

贝叶斯优化

贝叶斯优化（求解最小化问题）

```
0 确定函数$\mu(\cdot)$和$k(\cdot,\cdot)$用于描述高斯过程
1 随机选择$m_0$个解$\boldsymbol{x}$作为初始测试解，得到相应$m_0$个$f(\boldsymbol{x})$函数值
2 令$m \leftarrow m_0$
3 利用目前$m$个解的函数值$f(\boldsymbol{x}_1), \dots, f(\boldsymbol{x}_m)$更新$f(\boldsymbol{x})$的高斯分布，得到$\tilde{\mu}(\boldsymbol{x})$与$\tilde{\sigma}(\boldsymbol{x})$
4 根据选择标准，选择下一个测试解$\boldsymbol{x}_{m+1}$
5 获得$f(\boldsymbol{x}_{m+1})$
6 若算法终止条件不满足，$m \leftarrow m+1$并返回3；否则，输出最终解
```

常用高斯过程模型：$\mu(\boldsymbol{x}) = 0, k(\boldsymbol{x}_1, \boldsymbol{x}_2) = \exp(-\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|_2^2)$

常用下一个测试解的选择标准：选择令$-\tilde{\mu}(\boldsymbol{x}) + \tilde{\sigma}(\boldsymbol{x})$最大的解

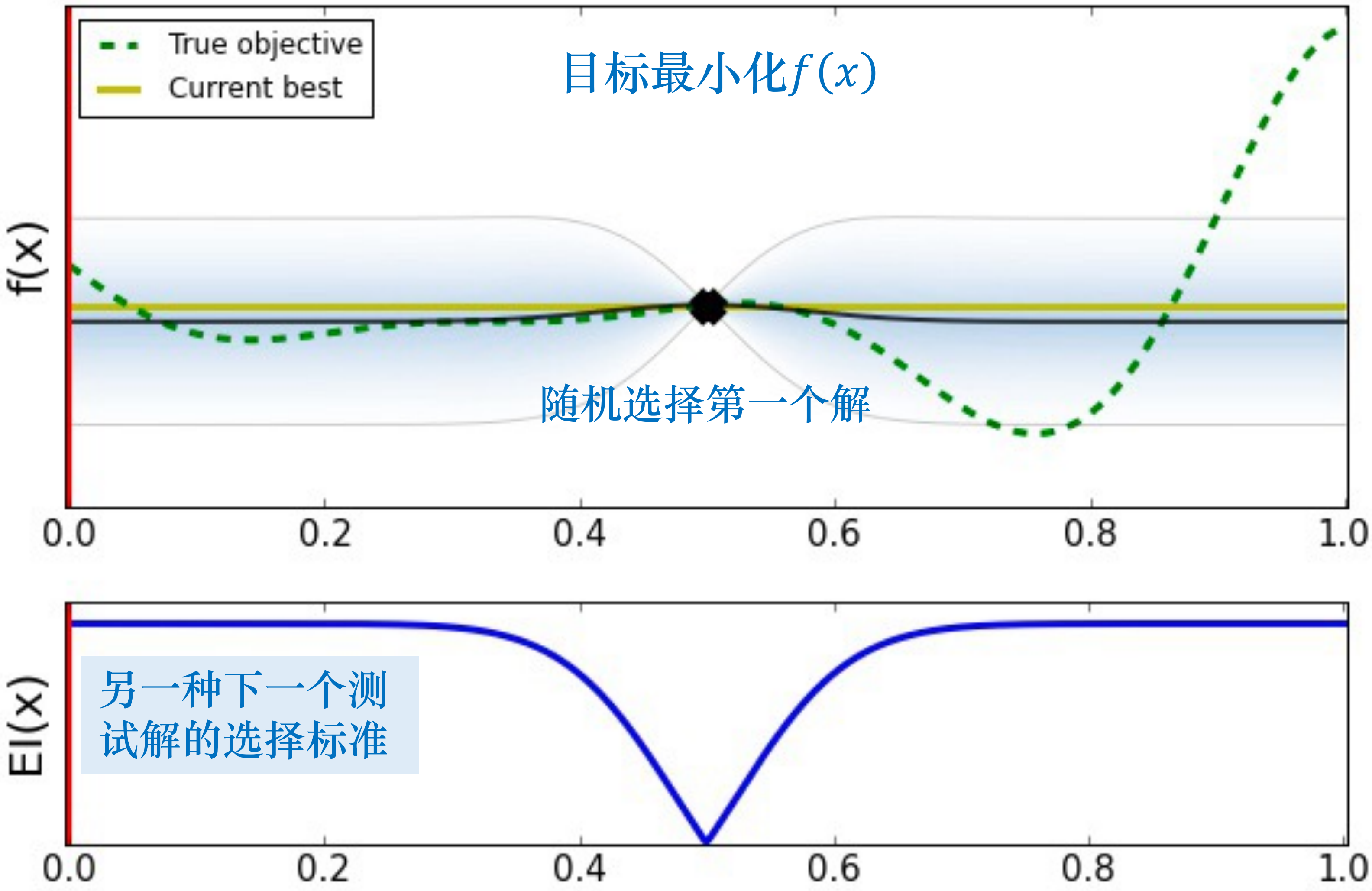
常用最终解的选择标准：选择已经测试的$m+1$个解中最好的解

贝叶斯优化

绿色曲线：真实的函数（不可见）

黑灰色带：解空间中各解函数值的高斯分布的均值、标准差

浅绿色直线：当前已搜索解中函数值的最小值



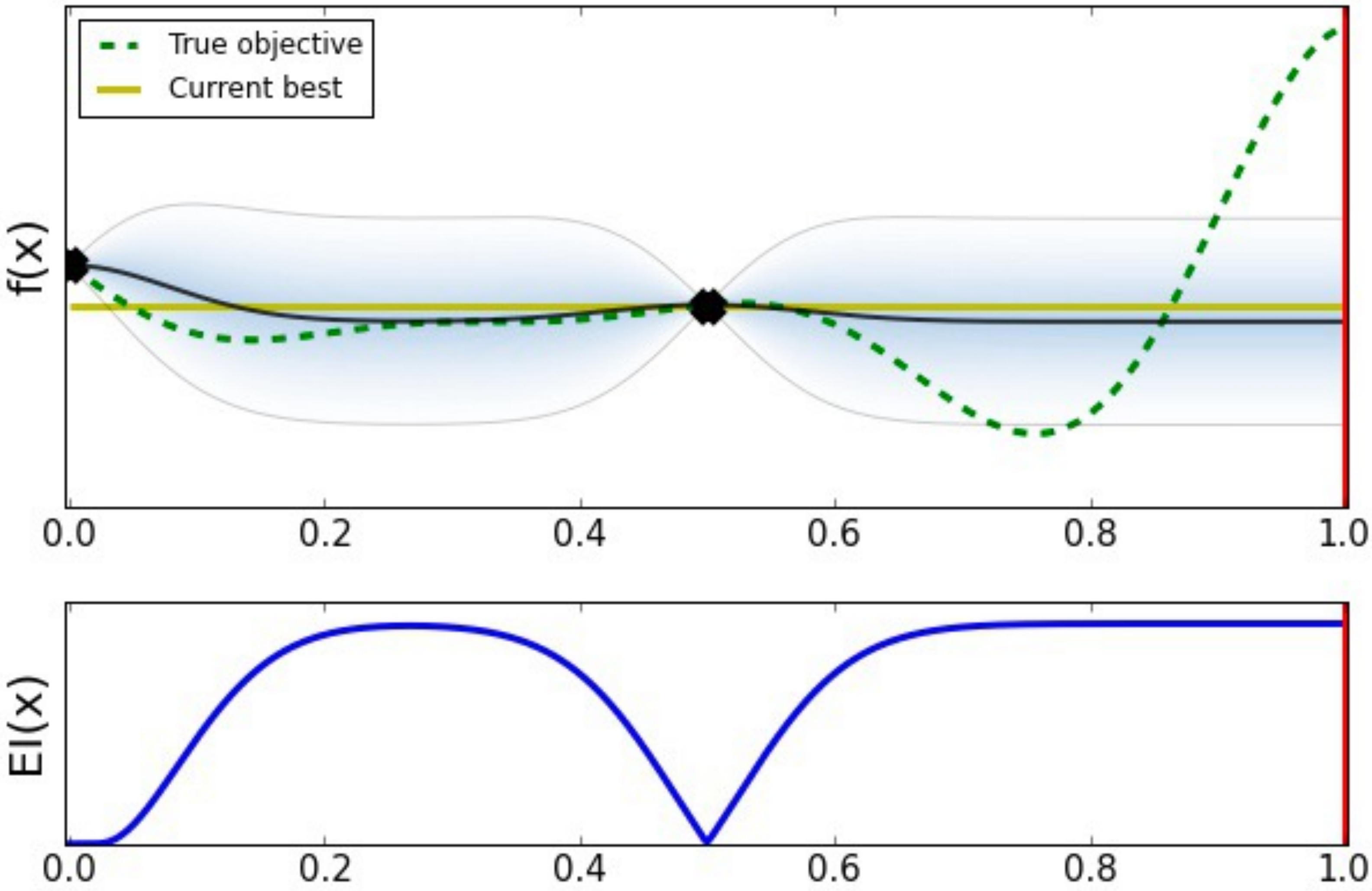
另一种下一个测试解的选择标准

贝叶斯优化

绿色曲线：真实的函数（不可见）

黑灰色带：解空间中各解函数值的高斯分布的均值、标准差

浅绿色直线：当前已搜索解中函数值的最小值

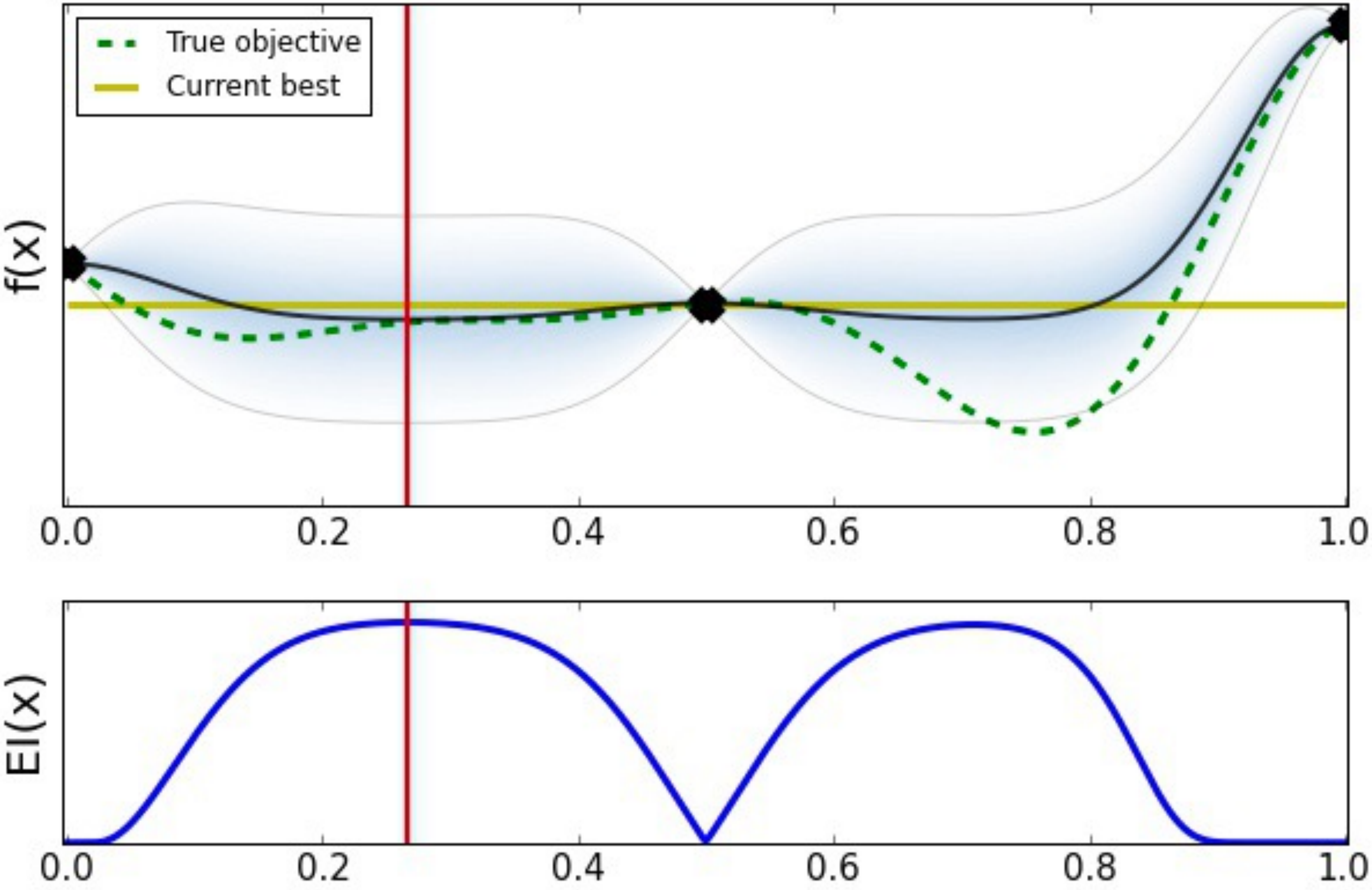


贝叶斯优化

绿色曲线：真实的函数（不可见）

黑灰色带：解空间中各解函数值的高斯分布的均值、标准差

浅绿色直线：当前已搜索解中函数值的最小值

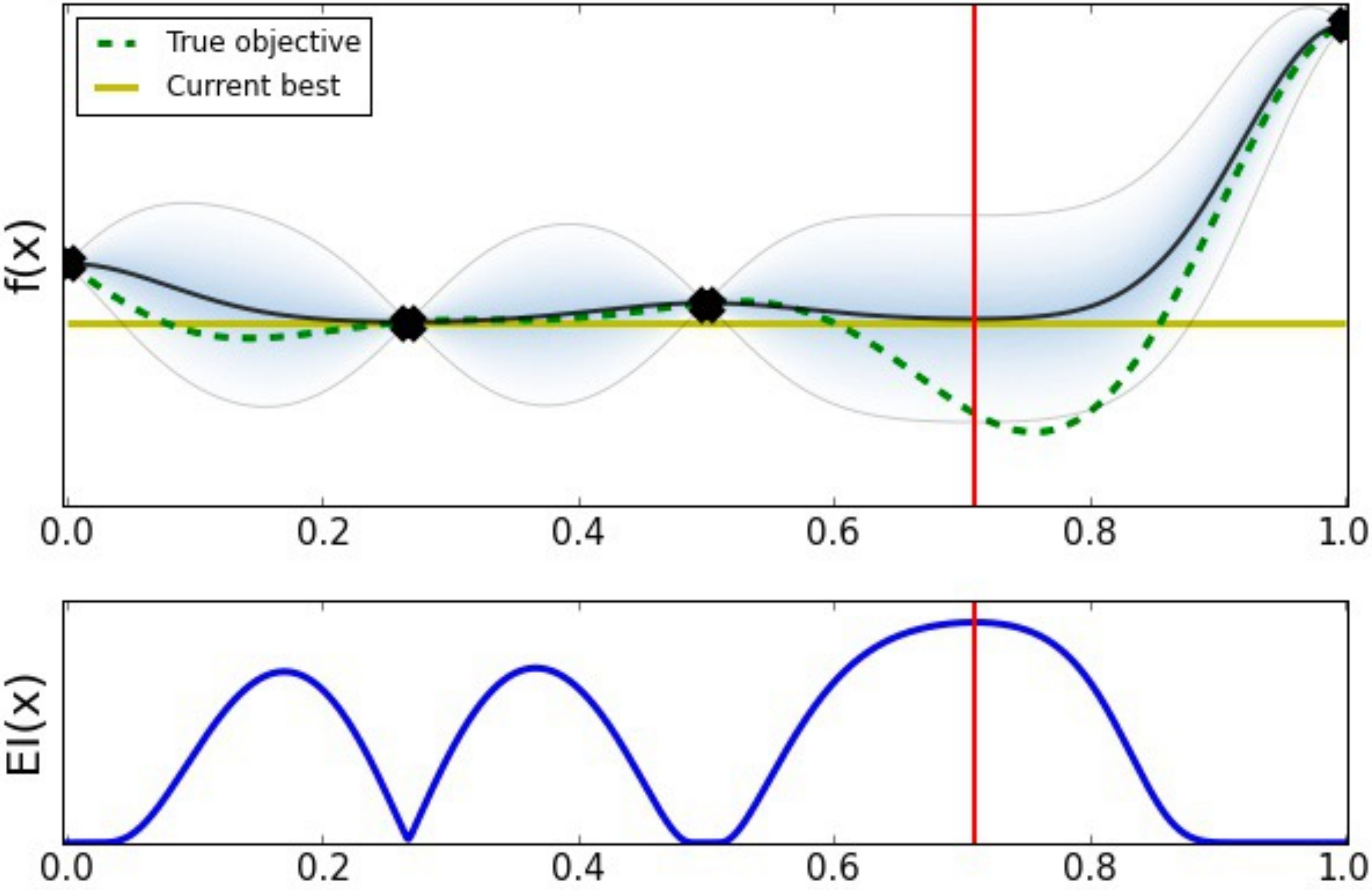


贝叶斯优化

绿色曲线：真实的函数（不可见）

黑灰色带：解空间中各解函数值的高斯分布的均值、标准差

浅绿色直线：当前已搜索解中函数值的最小值

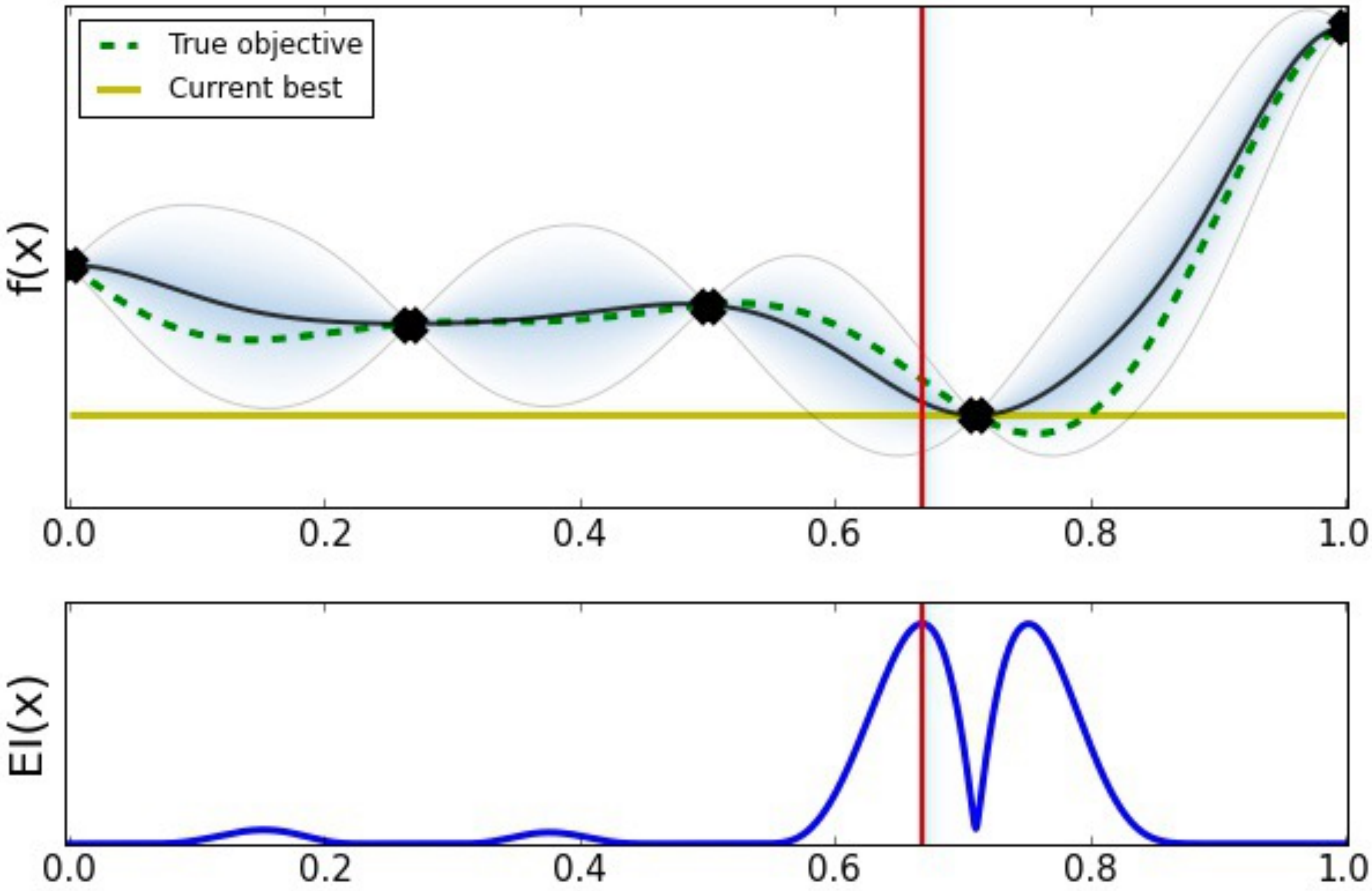


贝叶斯优化

绿色曲线：真实的函数（不可见）

黑灰色带：解空间中各解函数值的高斯分布的均值、标准差

浅绿色直线：当前已搜索解中函数值的最小值

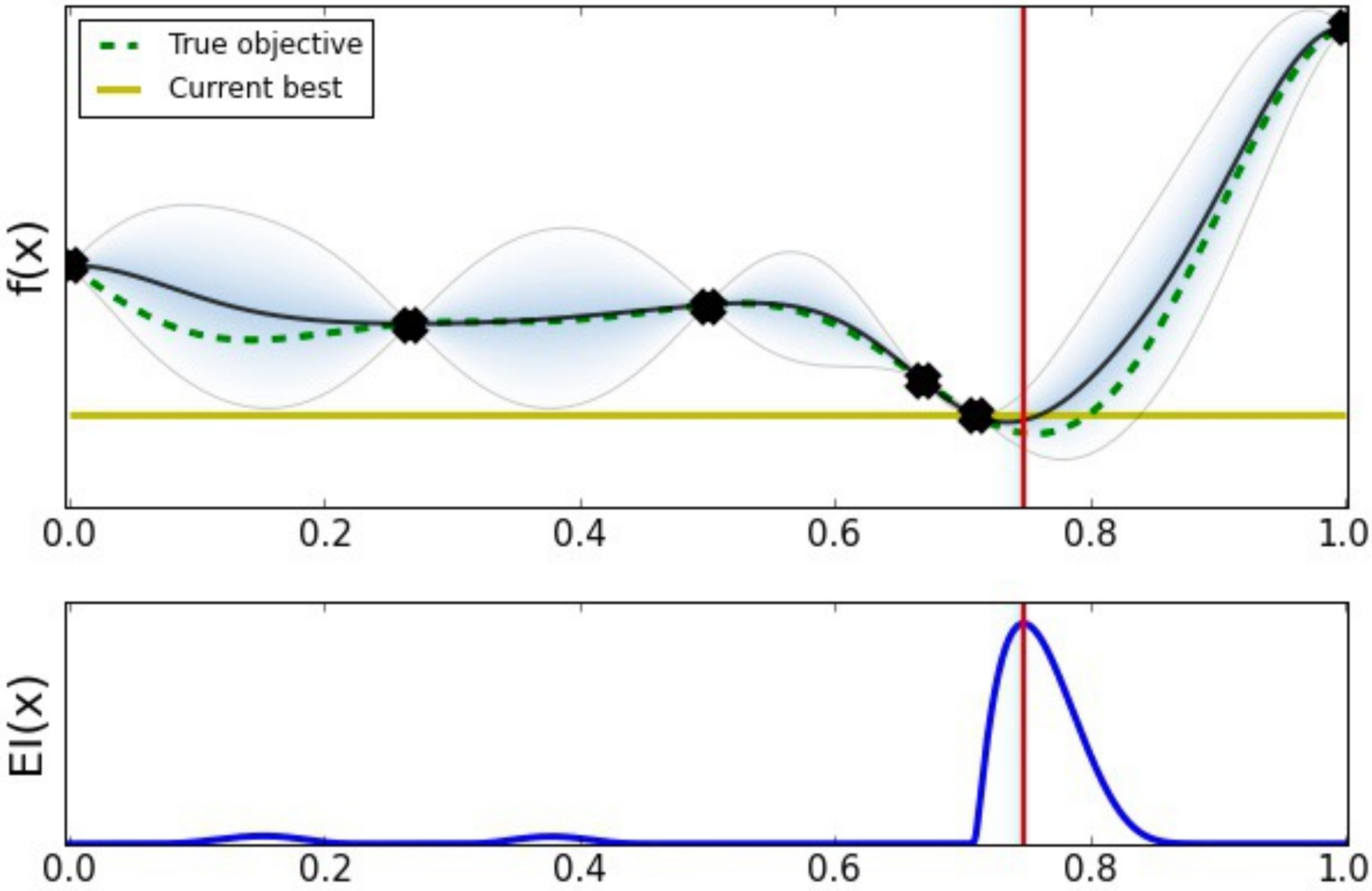


贝叶斯优化

绿色曲线：真实的函数（不可见）

黑灰色带：解空间中各解函数值的高斯分布的均值、标准差

浅绿色直线：当前已搜索解中函数值的最小值

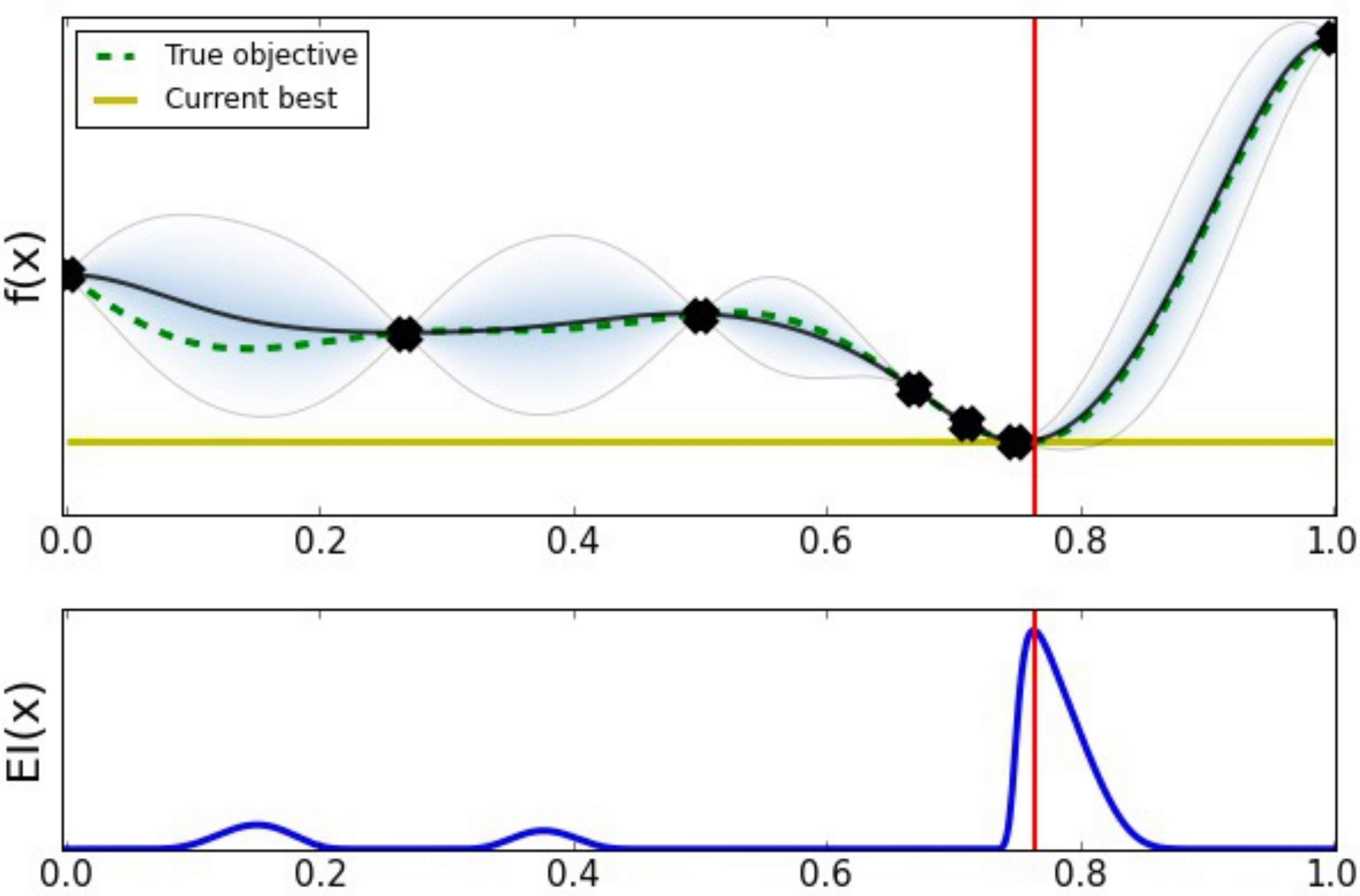


贝叶斯优化

绿色曲线：真实的函数（不可见）

黑灰色带：解空间中各解函数值的高斯分布的均值、标准差

浅绿色直线：当前已搜索解中函数值的最小值

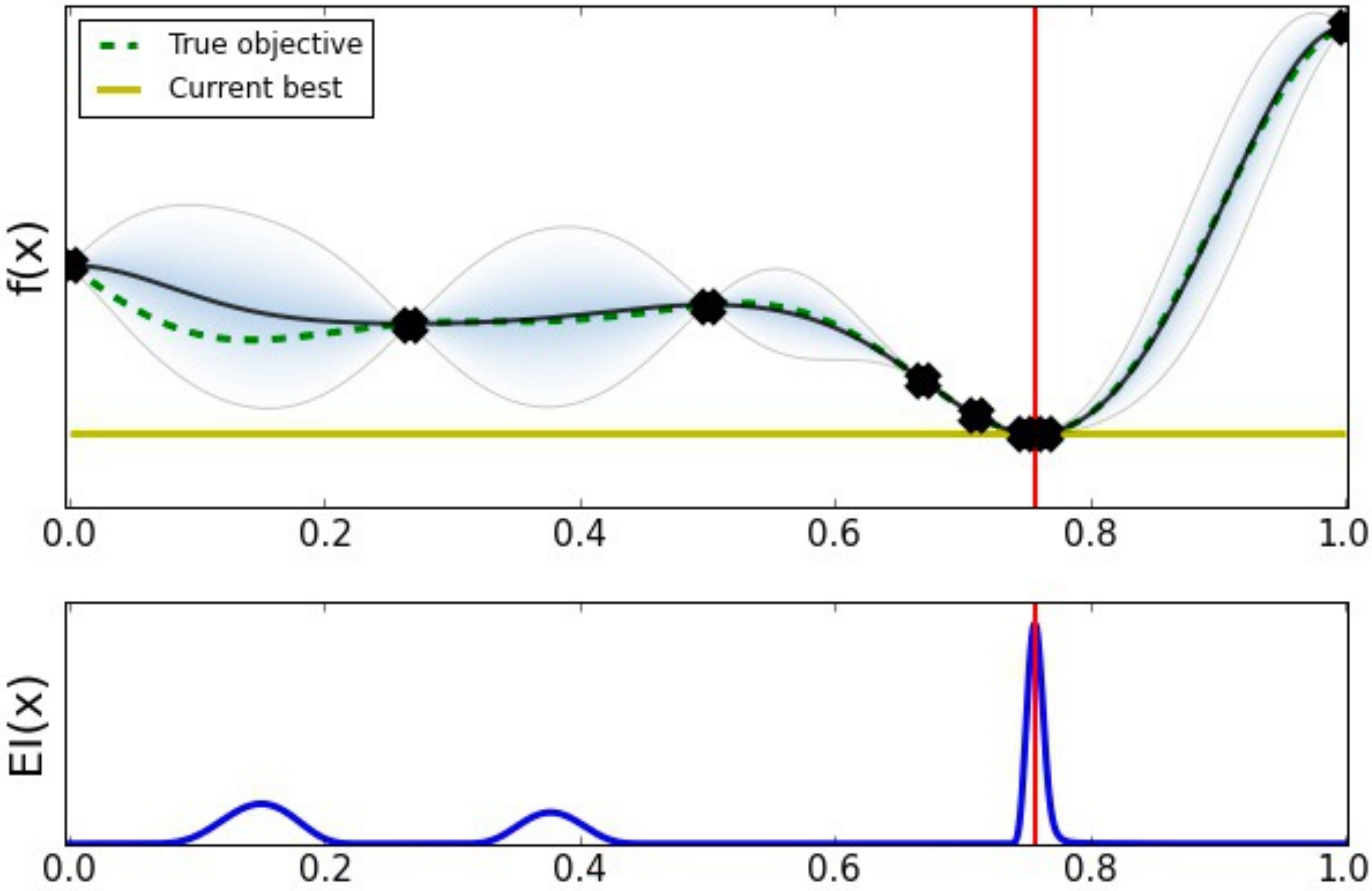


贝叶斯优化

绿色曲线：真实的函数（不可见）

黑灰色带：解空间中各解函数值的高斯分布的均值、标准差

浅绿色直线：当前已搜索解中函数值的最小值



贝叶斯优化

当采样点$\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots$很多时，计算$f(\boldsymbol{x})$的后验概率的$\tilde{\mu}(\boldsymbol{x})$与$\tilde{\sigma}(\boldsymbol{x})$的难度加大（如涉及高阶矩阵求逆运算）

思考：如果不用高斯过程对$f(\boldsymbol{x})$的形状建模，可否改用其它模型刻画?（如用线性回归拟合$f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), \ldots$）如何改进从而实现对$f(\boldsymbol{x})$的刻画?

贝叶斯优化

A Visual Exploration of Gaussian Processes

https://distill.pub/2019/visual-exploration-gaussian-processes/

后验概率计算结果

$$X|Y \sim \mathcal{N}(\,\mu_X + \Sigma_{XY}\Sigma_{YY}^{-1}(Y - \mu_Y),\ \Sigma_{XX} - \Sigma_{XY}\Sigma_{YY}^{-1}\Sigma_{YX}\,)$$
$$Y|X \sim \mathcal{N}(\,\mu_Y + \Sigma_{YX}\Sigma_{XX}^{-1}(X - \mu_X),\ \Sigma_{YY} - \Sigma_{YX}\Sigma_{XX}^{-1}\Sigma_{XY}\,)$$

可以尝试不同$k(\boldsymbol{x}_1, \boldsymbol{x}_2)$函数

RBF KERNEL	PERIODIC	LINEAR
$\sigma^2 \exp\left(-\frac{\|\|t-t'\|\|^2}{2l^2}\right)$	$\sigma^2 \exp\left(-\frac{2\sin^2(\pi\|t-t'\|/p)}{l^2}\right)$	$\sigma_b^2 + \sigma^2(t-c)(t'-c)$

本讲小结

- 进化算法之差分进化算法
- 群体智能之蚁群优化算法
- 群体智能之粒子群优化算法
- 贝叶斯优化

主要参考资料

Li et al. <A novel hybrid differential evolution algorithm with modified CoDE and JADE> Paper

S. Das and P. Suganthan <Differential Evolution: A Survey of the State-of-the-Art> Paper

Kelly Fleetwood <An Introduction to Differential Evolution> Slides

B站UP主 FancyZerOkami <蚂蚁模拟 Colony simulation> Video

Jeff Lettman <An Introduction to Ant colony optimization> Video

Joy Dutta <Ant colony optimization> Slides

C. Blum <Ant colony optimization: Introduction and recent trends> Paper

Krzysztof Schiff <Ant colony optimization algorithm for the 0-1 knapsack problem> Paper

Prakash Kotecha (IIT Guwahati) <Computer Aided Applied Single Objective Optimization> Slides

B站UP主 幼鹰me <粒子群算法可视化> Video

Nasser M. <Swarm Intelligence> Slides

Prakash Kotecha (IIT Guwahati) <Computer Aided Applied Single Objective Optimization> Slides

Javier Gonzalez (Lancaster University) <Introduction to Bayesian Optimization> Slides

Peter I. Frazier <A Tutorial on Bayesian Optimization> Paper

Jochen Görtler et al. <A Visual Exploration of Gaussian Processes> Webpage

谢谢！

参考内容
群体智能之人工蜂群算法

Part1 如何从复杂度的角度衡量算法好坏？

Part2 针对一些有特定形式的最优化问题，如何找到最优解？

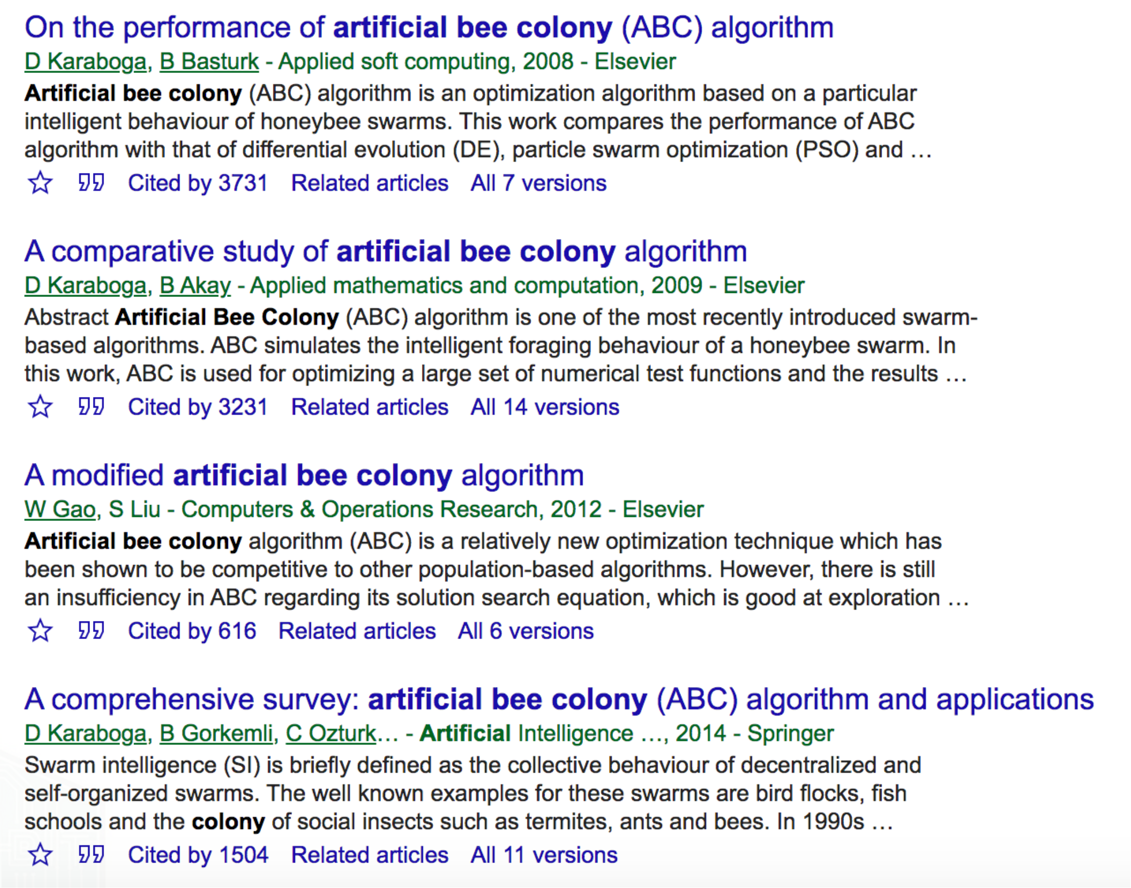
Part3 针对较复杂的最优化问题（如NP难问题），如何找到较好解？

局部搜索、模拟退火、遗传算法、差分进化算法、蚁群算法、粒子群优化算法、人工蜂群算法、贝叶斯优化

Part4 针对较复杂的多目标最优化问题，如何找到较好解？

人工蜂群算法

人工蜂群（Artificial Bee Colony，ABC）



On the performance of **artificial bee colony** (ABC) algorithm
D Karaboga, B Basturk - Applied soft computing, 2008 - Elsevier
Artificial bee colony (ABC) algorithm is an optimization algorithm based on a particular intelligent behaviour of honeybee swarms. This work compares the performance of ABC algorithm with that of differential evolution (DE), particle swarm optimization (PSO) and …
☆ ❞ Cited by 3731 Related articles All 7 versions

A comparative study of **artificial bee colony** algorithm
D Karaboga, B Akay - Applied mathematics and computation, 2009 - Elsevier
Abstract **Artificial Bee Colony** (ABC) algorithm is one of the most recently introduced swarm-based algorithms. ABC simulates the intelligent foraging behaviour of a honeybee swarm. In this work, ABC is used for optimizing a large set of numerical test functions and the results …
☆ ❞ Cited by 3231 Related articles All 14 versions

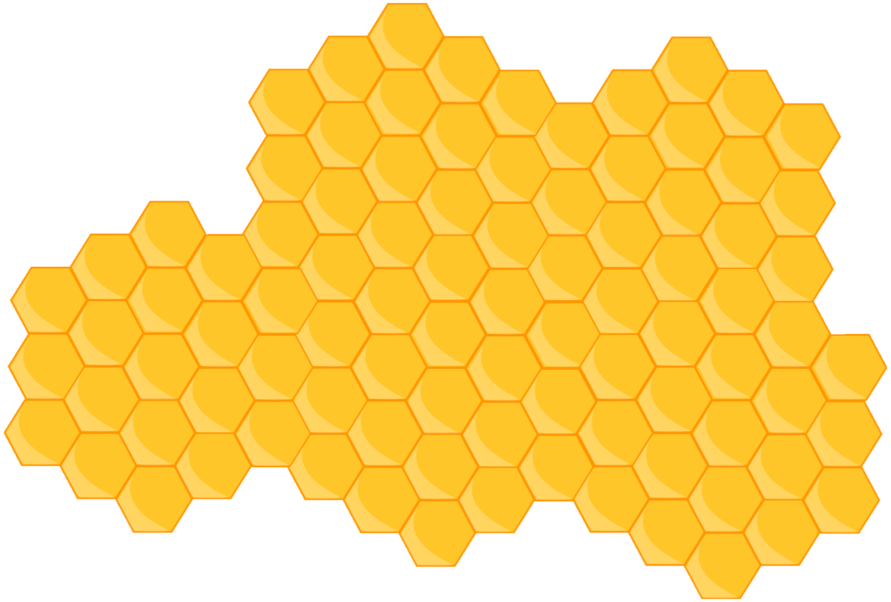
A modified **artificial bee colony** algorithm
W Gao, S Liu - Computers & Operations Research, 2012 - Elsevier
Artificial bee colony algorithm (ABC) is a relatively new optimization technique which has been shown to be competitive to other population-based algorithms. However, there is still an insufficiency in ABC regarding its solution search equation, which is good at exploration …
☆ ❞ Cited by 616 Related articles All 6 versions

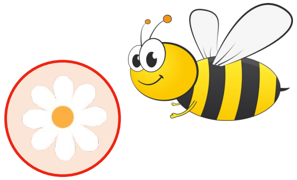
A comprehensive survey: **artificial bee colony** (ABC) algorithm and applications
D Karaboga, B Gorkemli, C Ozturk… - **Artificial** Intelligence …, 2014 - Springer
Swarm intelligence (SI) is briefly defined as the collective behaviour of decentralized and self-organized swarms. The well known examples for these swarms are bird flocks, fish schools and the **colony** of social insects such as termites, ants and bees. In 1990s …
☆ ❞ Cited by 1504 Related articles All 11 versions

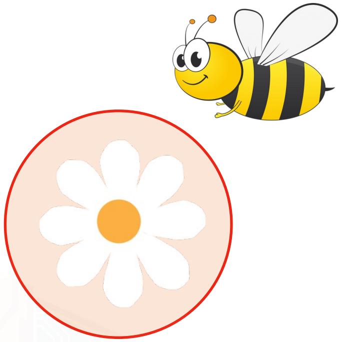
人工蜂群算法

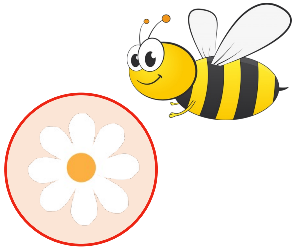
第一种蜂: 雇佣蜂 (employed bee)

模拟蜜蜂采蜜过程





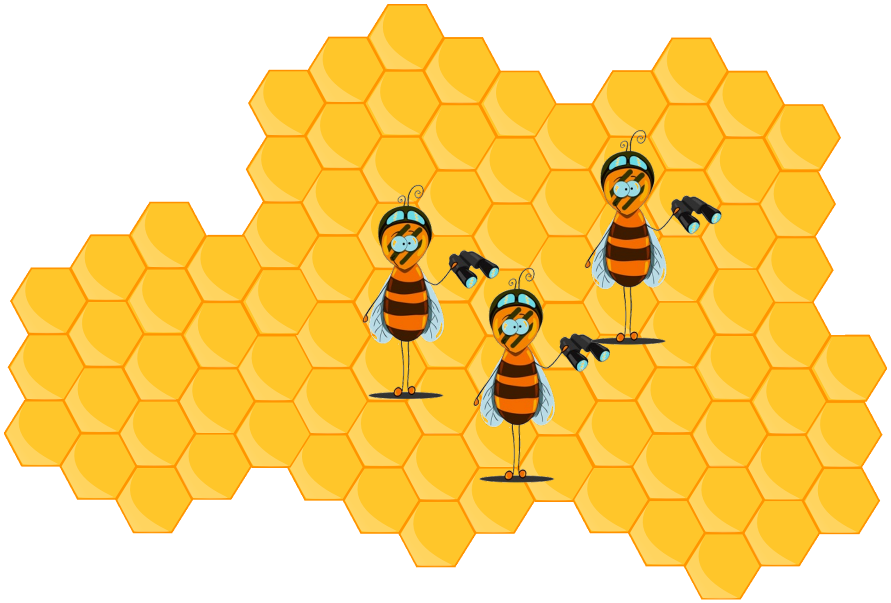


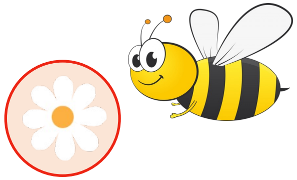


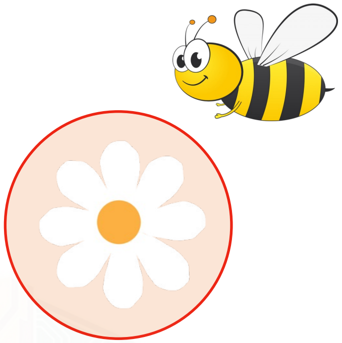
人工蜂群算法

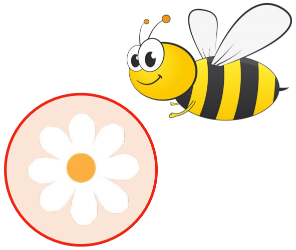
第二种蜂: 观望蜂 (onlooker bee)

模拟蜜蜂采蜜过程





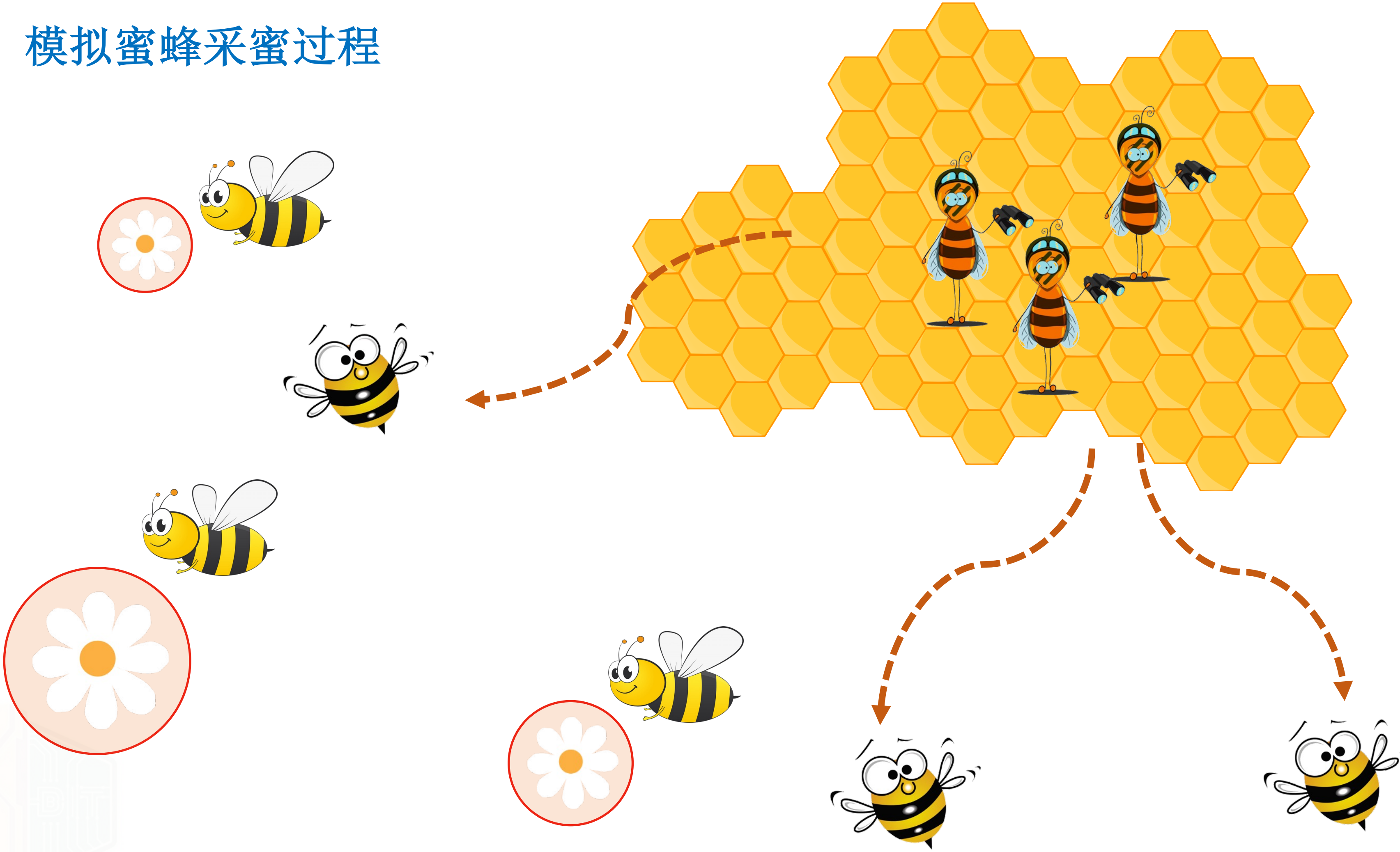




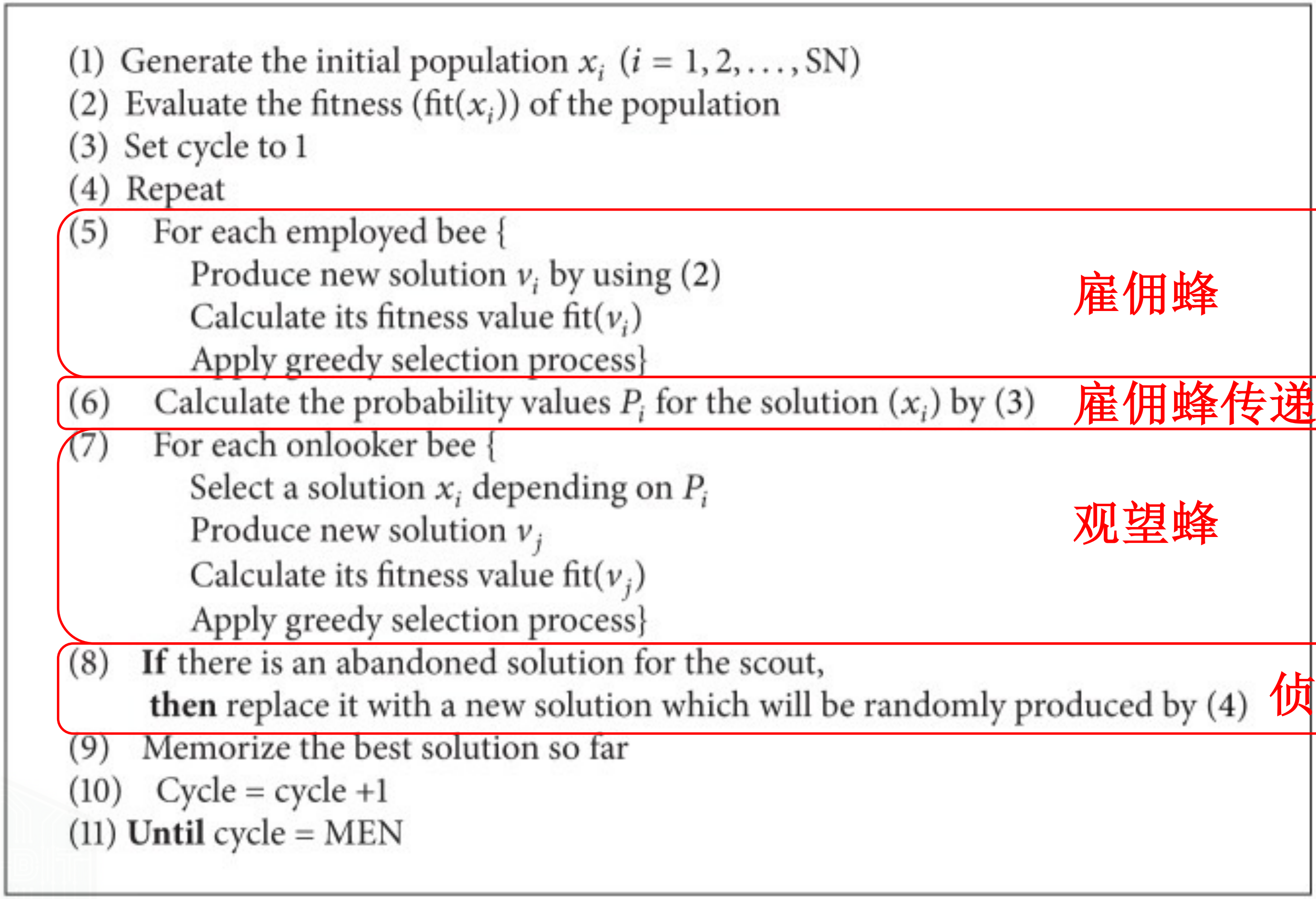
人工蜂群算法

第三种蜂: 侦察蜂 (scout bee)

模拟蜜蜂采蜜过程



人工蜂群算法



```
(1) Generate the initial population $x_i$ $(i = 1, 2, \ldots, SN)$
(2) Evaluate the fitness ($\mathrm{fit}(x_i)$) of the population
(3) Set cycle to 1
(4) Repeat
(5)     For each employed bee {
            Produce new solution $v_i$ by using (2)
            Calculate its fitness value $\mathrm{fit}(v_i)$
            Apply greedy selection process}
(6)     Calculate the probability values $P_i$ for the solution $(x_i)$ by (3)
(7)     For each onlooker bee {
            Select a solution $x_i$ depending on $P_i$
            Produce new solution $v_j$
            Calculate its fitness value $\mathrm{fit}(v_j)$
            Apply greedy selection process}
(8)     If there is an abandoned solution for the scout,
        then replace it with a new solution which will be randomly produced by (4)
(9)     Memorize the best solution so far
(10)    Cycle = cycle +1
(11) Until cycle = MEN
```

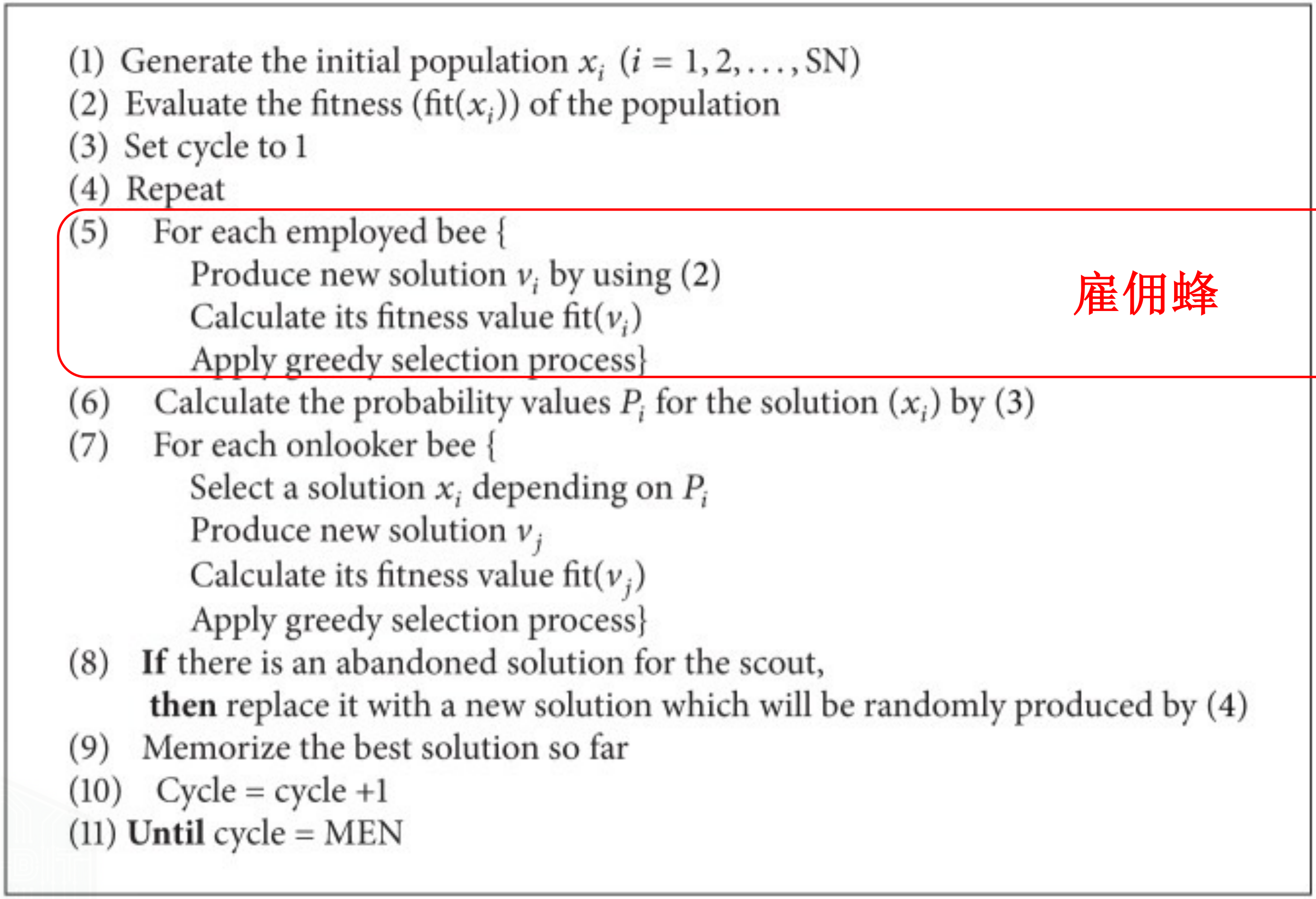
雇佣蜂

雇佣蜂传递信息

观望蜂

侦察蜂

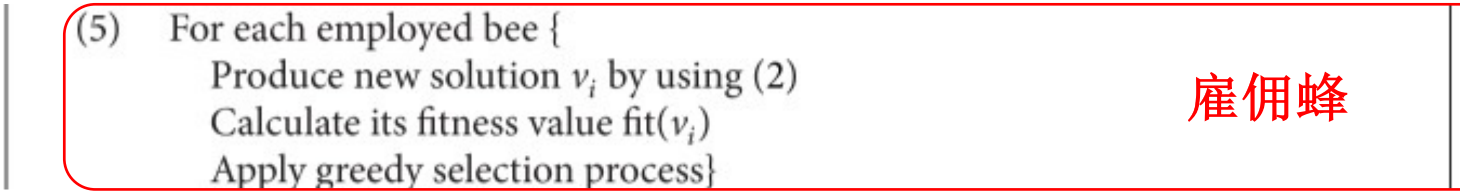
人工蜂群算法



```
(1)  Generate the initial population $x_i$ $(i = 1, 2, \ldots, \mathrm{SN})$
(2)  Evaluate the fitness ($\mathrm{fit}(x_i)$) of the population
(3)  Set cycle to 1
(4)  Repeat
(5)      For each employed bee {
             Produce new solution $v_i$ by using (2)
             Calculate its fitness value $\mathrm{fit}(v_i)$
             Apply greedy selection process}
(6)      Calculate the probability values $P_i$ for the solution $(x_i)$ by (3)
(7)      For each onlooker bee {
             Select a solution $x_i$ depending on $P_i$
             Produce new solution $v_j$
             Calculate its fitness value $\mathrm{fit}(v_j)$
             Apply greedy selection process}
(8)     If there is an abandoned solution for the scout,
         then replace it with a new solution which will be randomly produced by (4)
(9)     Memorize the best solution so far
(10)    Cycle = cycle +1
(11) Until cycle = MEN
```

雇佣蜂

人工蜂群算法

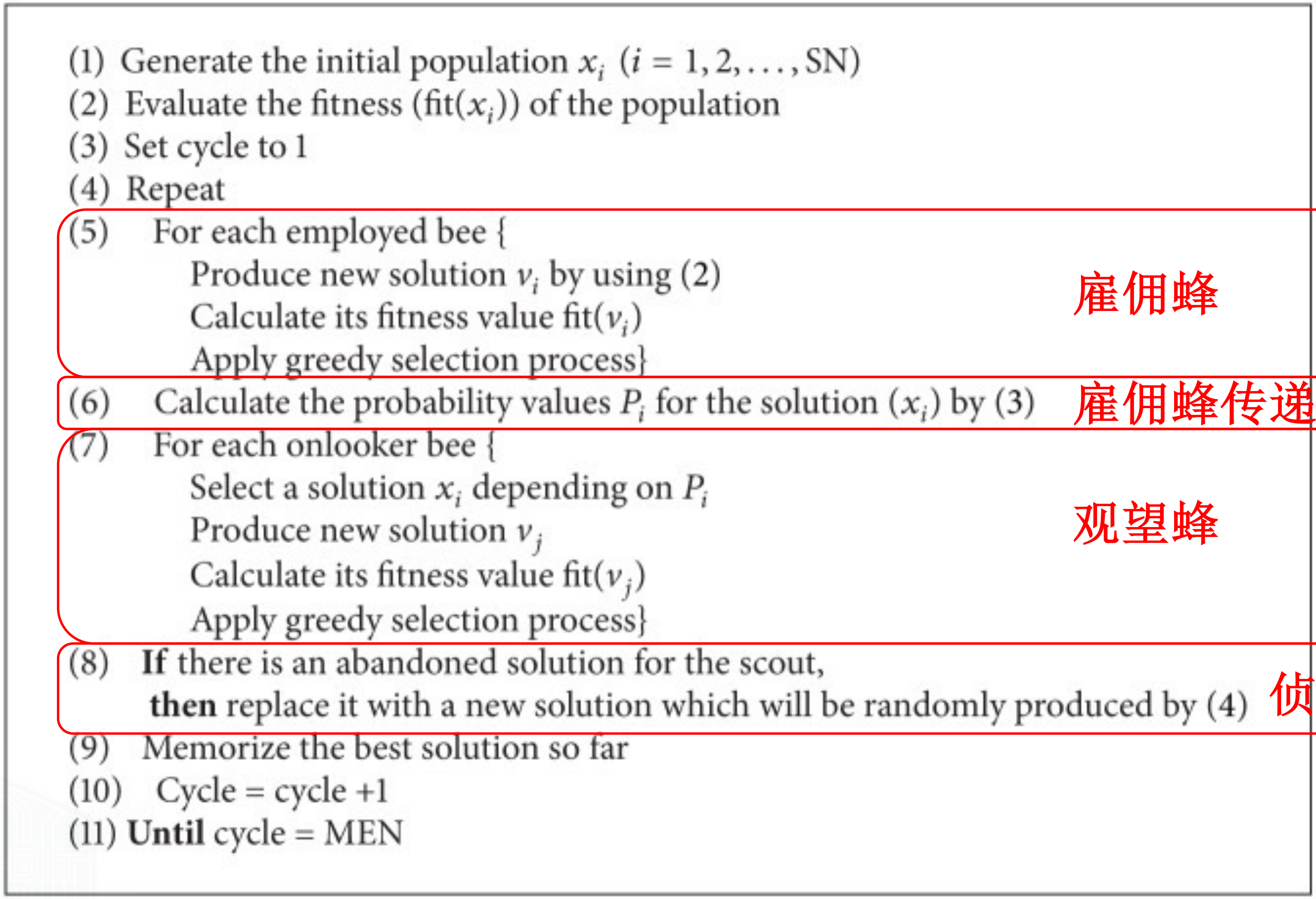


```
(5)    For each employed bee {
          Produce new solution $v_i$ by using (2)
          Calculate its fitness value fit($v_i$)
          Apply greedy selection process}
```

雇佣蜂

$$v_{ij} = \begin{cases} x_{ij} + \mathrm{rand} \cdot \left(x_{ij} - x_{kj} \right), & j = j_{\mathrm{rand}}, \\ x_{ij}, & j \neq j_{\mathrm{rand}}. \end{cases}$$

人工蜂群算法



```
(1)  Generate the initial population $x_i$ $(i = 1, 2, \ldots, SN)$
(2)  Evaluate the fitness (fit($x_i$)) of the population
(3)  Set cycle to 1
(4)  Repeat
(5)     For each employed bee {
            Produce new solution $v_i$ by using (2)
            Calculate its fitness value fit($v_i$)
            Apply greedy selection process}
(6)     Calculate the probability values $P_i$ for the solution ($x_i$) by (3)
(7)     For each onlooker bee {
            Select a solution $x_i$ depending on $P_i$
            Produce new solution $v_j$
            Calculate its fitness value fit($v_j$)
            Apply greedy selection process}
(8)    If there is an abandoned solution for the scout,
        then replace it with a new solution which will be randomly produced by (4)
(9)    Memorize the best solution so far
(10)    Cycle = cycle +1
(11) Until cycle = MEN
```

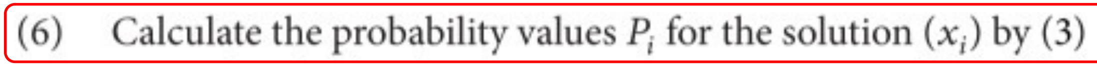
雇佣蜂

雇佣蜂传递信息

观望蜂

侦察蜂

人工蜂群算法

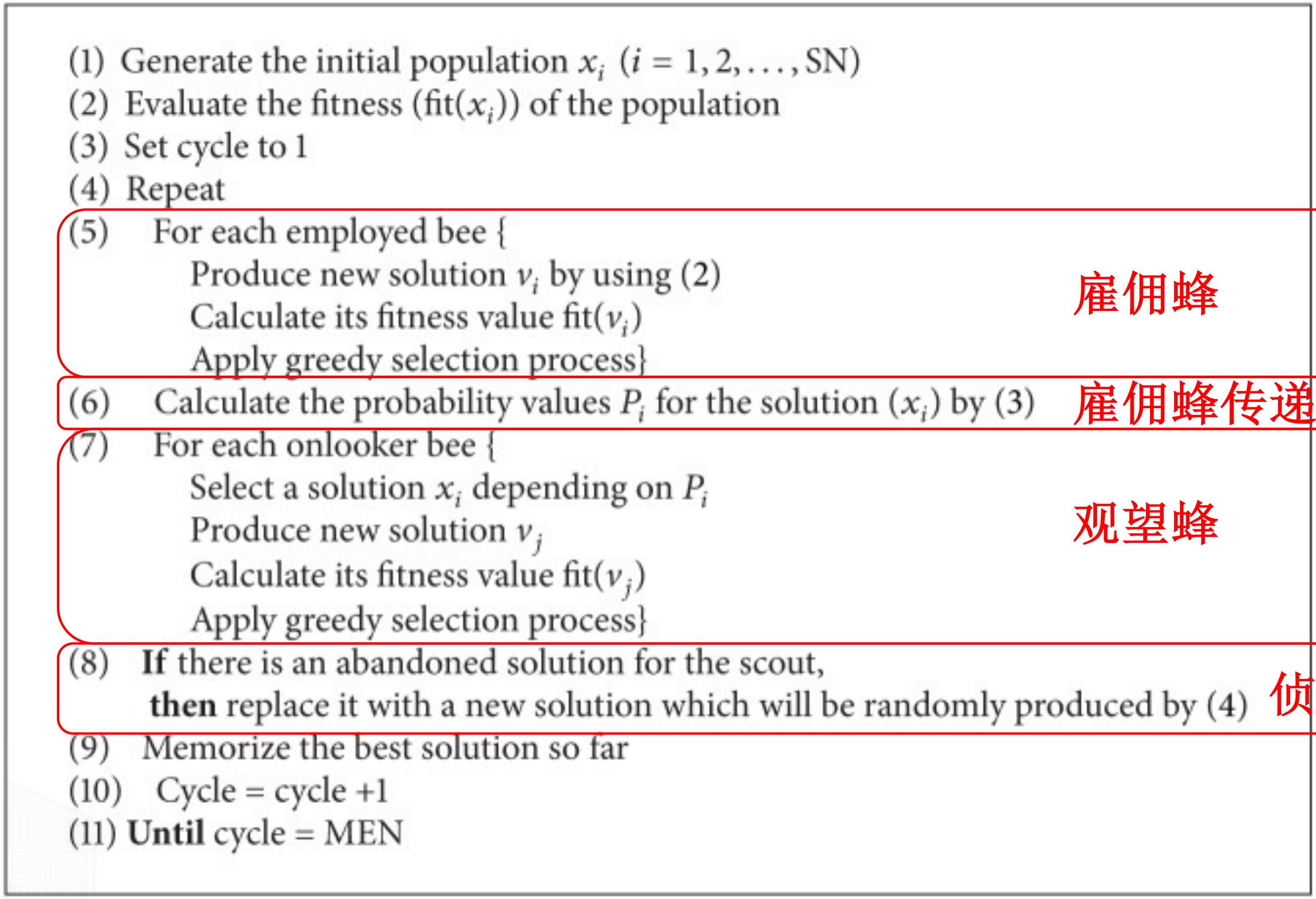


(6) Calculate the probability values P_i for the solution (x_i) by (3)

雇佣蜂传递信息

$$P_i = \frac{f_i}{\sum_{k=1}^{SN} f_k}$$

人工蜂群算法



```
(1) Generate the initial population $x_i$ $(i = 1, 2, \ldots, SN)$
(2) Evaluate the fitness ($\text{fit}(x_i)$) of the population
(3) Set cycle to 1
(4) Repeat
(5)     For each employed bee {
            Produce new solution $v_i$ by using (2)
            Calculate its fitness value $\text{fit}(v_i)$
            Apply greedy selection process}
(6)     Calculate the probability values $P_i$ for the solution $(x_i)$ by (3)
(7)     For each onlooker bee {
            Select a solution $x_i$ depending on $P_i$
            Produce new solution $v_j$
            Calculate its fitness value $\text{fit}(v_j)$
            Apply greedy selection process}
(8)     If there is an abandoned solution for the scout,
        then replace it with a new solution which will be randomly produced by (4)
(9)     Memorize the best solution so far
(10)    Cycle = cycle +1
(11) Until cycle = MEN
```

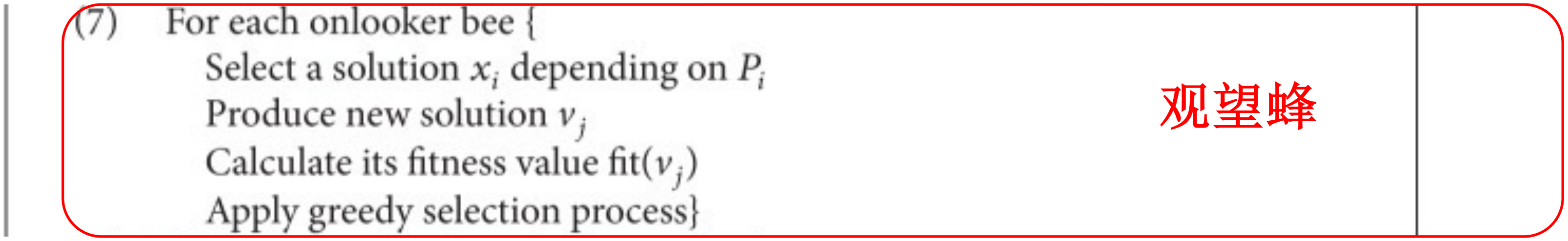
雇佣蜂

雇佣蜂传递信息

观望蜂

侦察蜂

人工蜂群算法



```
(7)  For each onlooker bee {
         Select a solution $x_i$ depending on $P_i$
         Produce new solution $v_j$
         Calculate its fitness value fit($v_j$)
         Apply greedy selection process}
```

观望蜂

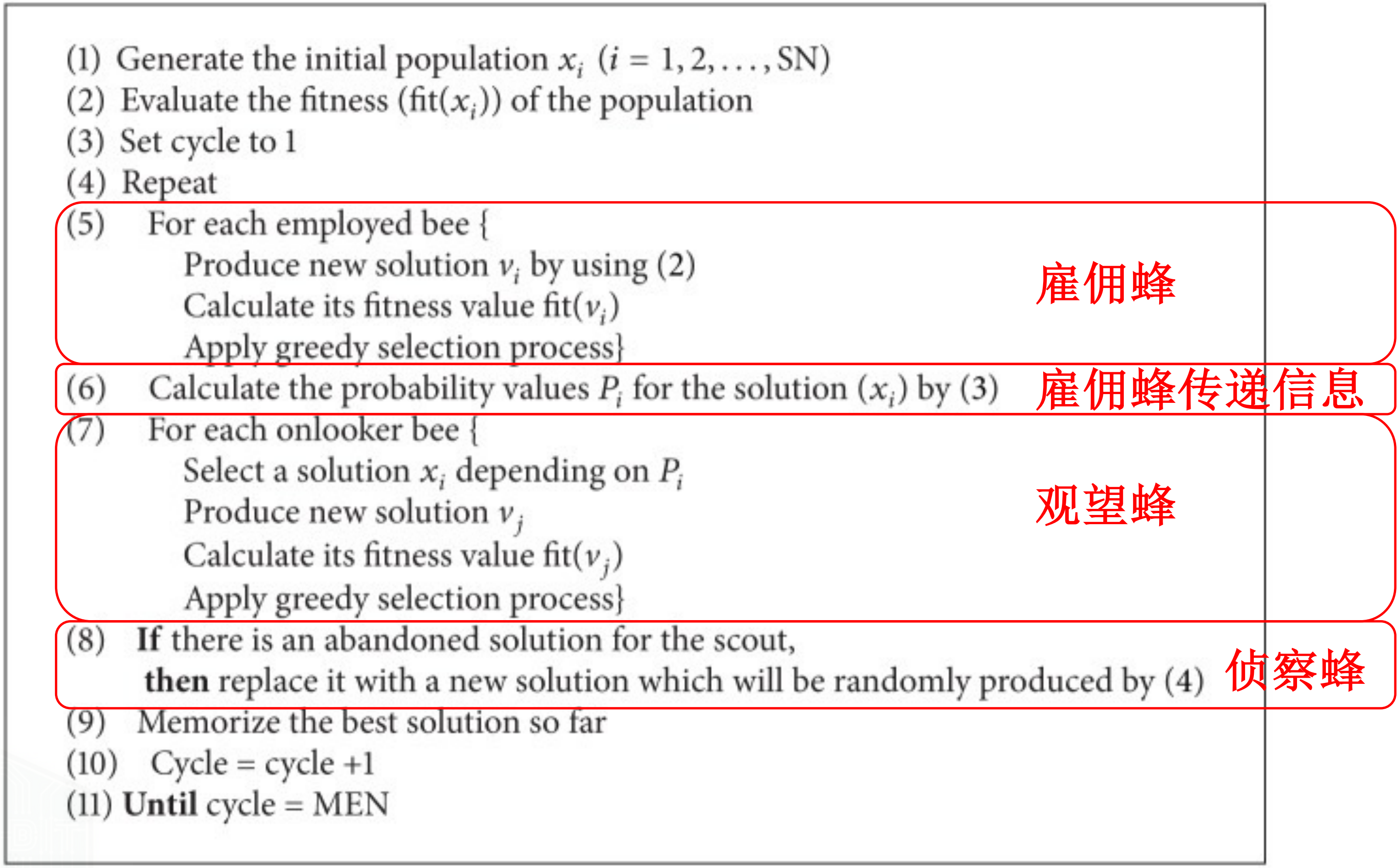
$$P_i = \frac{f_i}{\sum_{k=1}^{SN} f_k}$$

$$v_{ij} = \begin{cases} x_{ij} + \text{rand} \cdot \left(x_{ij} - x_{kj} \right), & j = j_{\text{rand}}, \\ x_{ij}, & j \neq j_{\text{rand}}. \end{cases}$$

对于最大化问题，直接根据每个解的目标函数值，按比例计算概率

对于最小化问题，需要先根据目标函数值进行变换，然后再计算概率（见后面例子）

人工蜂群算法



```
(1)  Generate the initial population $x_i$ $(i = 1, 2, \ldots, SN)$
(2)  Evaluate the fitness ($\mathrm{fit}(x_i)$) of the population
(3)  Set cycle to 1
(4)  Repeat
(5)    For each employed bee {
           Produce new solution $v_i$ by using (2)
           Calculate its fitness value $\mathrm{fit}(v_i)$
           Apply greedy selection process}
(6)    Calculate the probability values $P_i$ for the solution ($x_i$) by (3)
(7)    For each onlooker bee {
           Select a solution $x_i$ depending on $P_i$
           Produce new solution $v_j$
           Calculate its fitness value $\mathrm{fit}(v_j)$
           Apply greedy selection process}
(8)    If there is an abandoned solution for the scout,
       then replace it with a new solution which will be randomly produced by (4)
(9)    Memorize the best solution so far
(10)   Cycle = cycle +1
(11) Until cycle = MEN
```

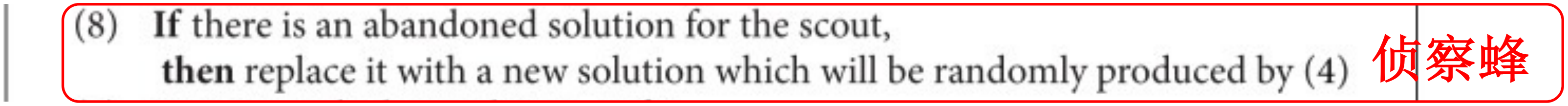
雇佣蜂

雇佣蜂传递信息

观望蜂

侦察蜂

人工蜂群算法



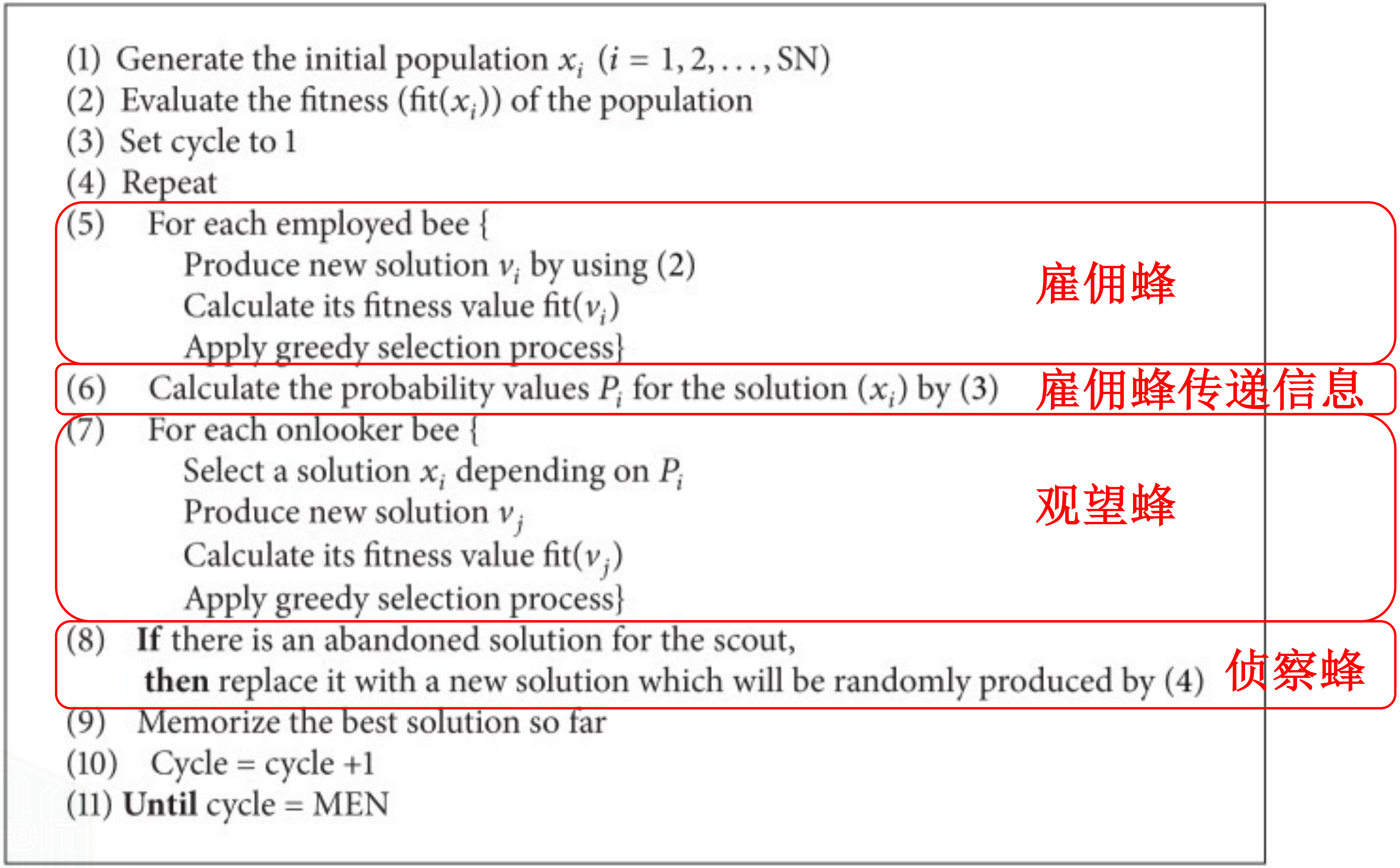
(8) **If** there is an abandoned solution for the scout,
then replace it with a new solution which will be randomly produced by (4)

侦察蜂

如果某一个解经过多轮一直没有改进，则随机产生新解将其代替

该过程可能把质量高的解替换掉

人工蜂群算法



```
(1) Generate the initial population $x_i$ $(i = 1, 2, \ldots, \mathrm{SN})$
(2) Evaluate the fitness ($\mathrm{fit}(x_i)$) of the population
(3) Set cycle to 1
(4) Repeat
(5)     For each employed bee {
            Produce new solution $v_i$ by using (2)
            Calculate its fitness value $\mathrm{fit}(v_i)$
            Apply greedy selection process}
(6)     Calculate the probability values $P_i$ for the solution $(x_i)$ by (3)
(7)     For each onlooker bee {
            Select a solution $x_i$ depending on $P_i$
            Produce new solution $v_j$
            Calculate its fitness value $\mathrm{fit}(v_j)$
            Apply greedy selection process}
(8)     If there is an abandoned solution for the scout,
        then replace it with a new solution which will be randomly produced by (4)
(9)     Memorize the best solution so far
(10)    Cycle = cycle +1
(11) Until cycle = MEN
```

雇佣蜂

雇佣蜂传递信息

观望蜂

侦察蜂