浙江大学
Zhejiang University

生物智能与算法
课程报告

题　　目　人工鱼群算法

学生姓名　罗晓倩

学　　号　21721107

学　　院　计算机科学与技术

2018 年 **4** 月 **12** 日

摘　要

人工鱼群算法是于 2002 年，由李晓磊等人提出的一种基于动物自治体的寻优算法。是一种根据鱼群的行为特点，应用动物自治体模型，自上而下设计的新型寻优策略。本报告主要介绍人工鱼群算法的原理和描述，算法的性能分析以及特定的问题解决。

关键字：　人工鱼群　优化　动物自治体

目　录

1 算法背景

提出者：李晓磊等人，2002 年。

提出背景：在动物的进化过程中, 经过漫长的自然界的优胜劣汰, 形成了形形色色的觅食和生存方式, 这些方式为人类解决问题的思路带来了不少启发和鼓舞。动物一般不具有人类所具有的复杂逻辑推理能力和综合判断能力的高级智能, 它们的目的是在个体的简单行为或通过群体的简单行为而达到或突现出来的。

动物行为具有以下几个特点: 1) 适应性: 动物通过感觉器官来感知外界环境, 并应激性的做出各种反应, 从而影响环境, 表现出与环境交互的能力; 2) 自治性: 动物有其特有的某些行为, 在不同的时刻和不同的环境中能够自主的选取某种行为, 而不是通过外界的控制或指导; 3) 盲目性: 不像传统的基于知识的智能系统, 有着明确的目标; 单个个体的行为是独立的, 与总目标之间没有直接的关系; 4) 突现性: 总目标的完成是在个体行为的运动过程中突现出来的; 5) 并行性: 各个体的行为是实时的、并行进行的。

随着人工智能和人工生命的兴起，出现了一些仿生算法，包括蚁群算法和粒子群算法等。将人工智能的思想应用于问题的寻优，实施一些高级的计算方法。应用动物自治体的模式来定义实体，让他们在问题空间中自主的活动，从而达到解决问题的目的。

2 算法简介

2.1 鱼群模式

2.1.1 视觉

虚拟人工鱼实体的当前状态为 X，Visual 为其视野范围，状态 X_v 为其在某时刻视点所在的位置，如果该位置的状态优于当前状态，则考虑向该位置方向前进一步，即到达状态 X_{next}

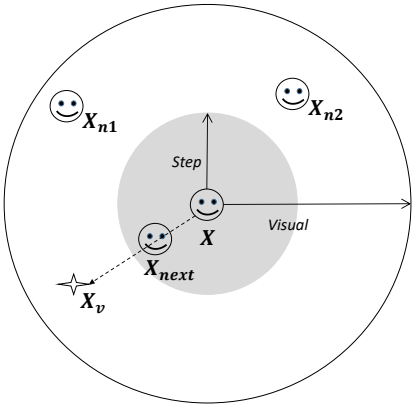


图 1: 人工鱼的视野和移动步长

2.1.2 鱼群行为分析

在一片水域中, 鱼生存数目最多的地方一般就是本水域中富含营养物质最多的地方, 因此, 该算法就依据这一特点来模仿鱼群的觅食行为从而实现寻优。提出者通过对鱼类生活习

性的观察, 总结并提取出了适用于该算法的以下几种典型的行为:

- 聚群行为：鱼类进化过程中的一种生存方式，大量或少量的鱼都能聚集成群，进行集体觅食和躲避敌害。
- 追尾行为：当某一条鱼或几条鱼发现食物时，它们附近的鱼会尾随其后快速游过来，进而导致更远处的鱼也尾随过来。
- 随机行为：鱼在水中悠闲的自由游动，基本上是随机的，其视也是为了更大范围地寻觅食物或同伴。
- 觅食行为：生物的基本行为，趋向食物的一种活动。一般可以认定它是同过视觉或味觉来感知水中的食物量或浓度来选择趋向。

2.1.3 人工鱼

人工鱼是真实鱼个体的一个虚拟实体，用来进行问题的分析和说明。
如图所示，可以将人工鱼看作是一个封装了自身数据星系和一系列行为的一个实体，同过感官来接受环境的刺激信息，并通过控制尾鳍来做出相应的应激活动。
人工鱼所在的环境主要是问题的解空间和其他人工鱼的状态，它在下一刻的行为取决于目前自身状态和目前环境的状态，并且它还通过自身活动的同时来影响环境，进而影响其他同伴的活动。

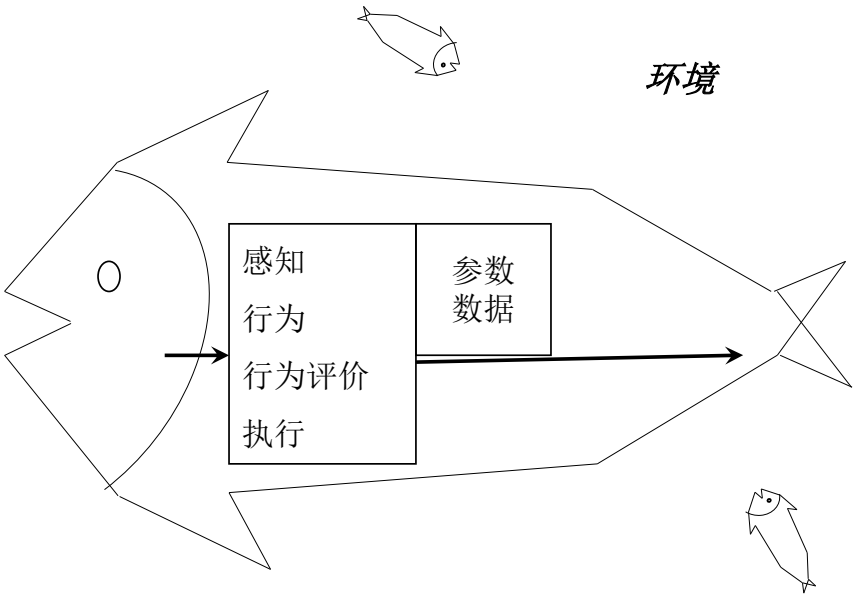


图 2: 人工鱼实体

2.1.4 问题解决

实际问题的解决是通过自治体在自主的活动过程中以某种形式表现出来的。在一片水域中，鱼生存的数目最多的地方一般就是本水域中富含营养物质最多的地方，依据这一特点来模仿鱼群的觅食等行为，从而实现全局寻优，这就是鱼群算法的基本思想。
在寻优过程中，通常会有两种方式表现出来：

- 一种形式是通过人工鱼最终的分布情况来确定最优解的分布，通常随着寻优解过程的进展，人工鱼往往会聚集在极值点的周围，而且，全局最优的极值点周围能聚集较多的人工鱼
- 另一种形式是在人工鱼的个体状态之中表现出来的，即在寻优的过程中，跟踪记录最优个体的状态，类似于遗传算法等的方式

2.2 人工鱼模型

```
class Artificial_fish
{
Various:
    float AF_X[n];                  //AFs position
    float AF_step;                  //the distance that AF can move for each step
    float AF_visual;                //the visual distance of AF
    float try _number               //attempt time in the behacior of prey
    float delta;                    //the condition of jamming
Functions:
    float AF_foodconsistence();     //the food consistence of AFs current position
    void AF_move();                 //AF move to the next position
    float AF_follow();              //the behavior of follow
    float AF_prey();                //the behavior of prey
    float AF_swarm();               //the behavior of swarm
    int AF_evaluate();              //evaluate and select the behavior
    void AF_init();                 //to initialize the AF
    Artificial_fish();
    virtual Artificial_fish;
};
```

一些定义:Step: 人工鱼移动最大步长;δ : 拥挤度因子;Visual: 人工鱼感知距离, 个体状态:$(x_1, x_2, \cdots, x_n)$, 其中 x_i 为欲寻优的变量；Y=f(X) 表示当前所在位置食物浓度, 其中 Y 为目标函数值；人工鱼个体之间的距离：$d_{i,j} = |X_i - X_j|$

2.3 行为描述

2.3.1 觅食行为

设人工鱼当前状态为 X_i，在其感知范围内随机选择一个状态 X_j，如果在求极大问题中，$Y_i < Y_j$(或在求极小问题中，$Y_i > Y_j$，因极大和极小问题可以互相转换，所以以下均以求极大问题讨论)，则向该方向前进一步；反之，再重新随机选择状态 X_j，判断是否满足前进条件；反复几次后，如果仍不满足前进条件，则随机移动一步。伪代码描述如下：

```
float Aritificial_fish::AF_prey()
{
    for(i = 0; i < try_number; i++)
    {
        X_j = X_i + Rand() · Visual;
        if (Y_i < Y_j)
            X_{i|next} = X_i + Rand() · Step · (X_j − X_i)/‖X_j − X_i‖;
        else
            X_{i|next} = X_i + Rand() · Step;
    }
```

```
return AF_foodconsistence(X_{i|next});
}
```

2.3.2 聚群行为

设人工鱼当前状态为 X_i，探索当前邻域内（即 $d_{i,j} < Visable$）的伙伴数目 n_f 及中心位置 X_c，如果 $Y_c/n_f > \delta Y_i$，表明伙伴中心有较多的食物并且不太拥挤，则朝伙伴的中心位置方向前进一步；否则执行觅食行为。伪代码描述如下：

```
float Aritificial_fish::AF_swarm()
{
    n_f = 0; X_c = 0
    for(j = 0; i < friend_number; j++)
        if(d_{i,j} < Visual){n_f++; X_c += X_j;}
    X_c = X_c/n_f;
    if(Y_c/n_f > δY_i)
        X_{i|next} = X_i + Rand() · Step · (X_c − X_i)/‖X_c − X_i‖;
    else
        AF_prey();
return AF_foodconsistence(X_{i|next});
}
```

2.3.3 追尾行为

设人工鱼当前状态为 X_i，探索当前邻域内（即 $d_{i,j} < Visable$）的伙伴中 Y_i 为最大的伙伴 X_j，如果 $Y_j/n_f > \delta Y_i$，表明伙伴 X_j 的状态具有较高的食物浓度并且周围不太拥挤，则朝伙伴 X_j 的方向前进一步；否则执行觅食行为。伪代码描述如下

```
float Aritificial_fish::AF_follow()
{
    Y_max = −∞;
    for(j = 0; i < friend_number; j++)
        if(d_{i,j} < Visual&&Y_j > Y_max)
            {Y_max = Y_j; X_max = X_j;}
    n_f = 0
    for(j = 0; i < friend_number; j++)
        if(d_{max,j} < Visual){n_f++;}
    if(Y_max/n_f > δY_i)
        X_{i|next} = X_i + Rand() · Step · (X_max − X_i)/‖X_max − X_i‖;
    else
        AF_prey();
return AF_foodconsistence(X_{i|next});
}
```

2.3.4 随机行为

随机行为的实现即在视野中随机选择一个状态，然后向该方向移动，实际上就是觅食行为的一个缺省行为。

2.4 行为选择

根据所要解决的问题性质, 对人工鱼当前所处的环境进行评价, 从而选择一种行为。如对于求取极大值的问题, 最简单的评估方法可以用试探法, 就是模拟执行聚群、追尾等行为, 然后评价行动后的值, 选择其中的最大者来实际执行, 缺省的行为方式为觅食行为。

3 算法实现

3.1 算法描述

每个人工鱼探索它的环境状况（包括目标函数的变化情况和伙伴的变化情况），从而选择一种行为最终，人工鱼集结在几个局部极值的周围一般情况下，在讨论求极大问题时，拥有较大的 $AF\_foodconsistence$ 值的人工鱼一般处于值较大的极值域周围这有助于获取全局极值域，而值较大的极值区域周围一般能集结较多的人工鱼，这有助于判断并获取全局极值。

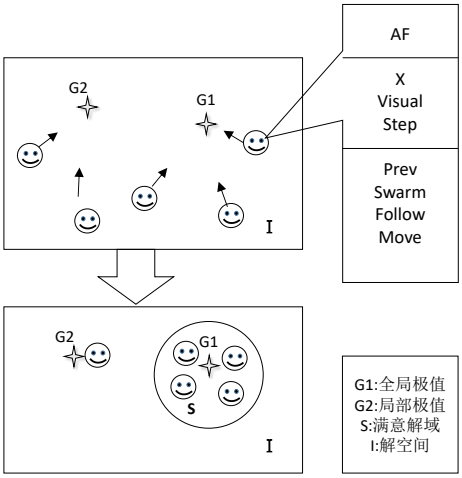


图 3: 算法示意图

Algorithm 1 AFA 算法

```
:: AF_init();
while the resulr is satisfied do
    switch(:: AF_evaluat())
      case  value1:
        :: AF_follow();
      case  value1:
        :: AF_swarm();
      default:
        :: AF_prey();
    end switch
    AF_move();
    get_result();
end while
```

3.2 仿真实验研究

3.2.1 仿真对象

寻优对象选取了下面的非线性目标函数：

$$maxf(x,y) = \frac{sin(x)}{x}\ \frac{siny}{y}$$
$$s.t.x \in [-10,10]$$
$$y \in [-10,10]$$

如图所示，可以看出，该非线性函数在全局极大值的周围密布着许多局部极值，通常的寻优算法极易陷入局部极值或在局部极值间震荡，比较适用于验证算法的性能。

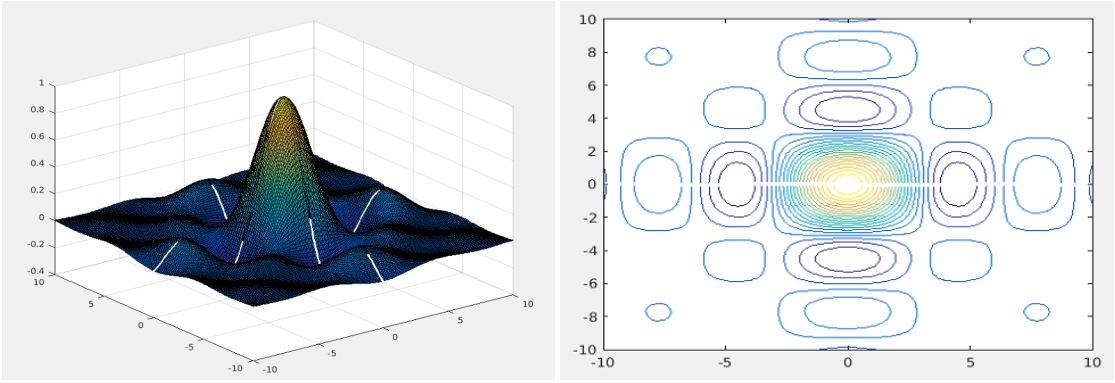


图 4: 具有多个局部极值的非线性目标函数

3.2.2 仿真结果

下面的仿真实验中，AF 的数量为 10 个，为了检验其寻优的效率，令它们的初始位置都处于距全局极值最远的（-10，10）处，其感知距离 Visual=2.5，每次移动的步长 Step=0.3，拥挤度因子 $\delta = 0.618$. 图 5 为 AF 只执行聚群行为的仿真结果，如图 5（a）中，AF 迅速聚集在最近的局部极值点附近，图 5（b）中已经有了 AF 到达全局极值的附近，图 5（c）中有更多的 AF 到达了全局极值点的附近，且有少数 AF 聚集到了其他的局部极值点附近，从寻优曲线可以看出，AF 能较快地跳出局部极值点，从而较快地找到全局极值。

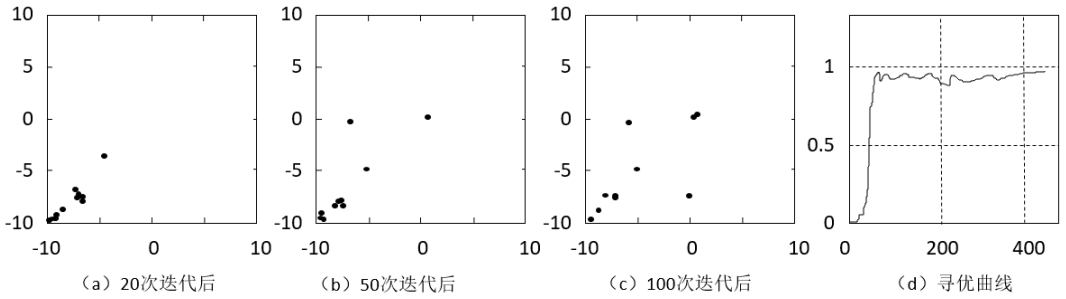


图 5: 聚群行为的仿真结果

图 6 为 AF 只执行追尾行为的仿真结果。如图 6（a）中，AF 迅速聚集在最近的局部极值点附近，图 6（b）中，AF 又迅速向附近的极值点聚集，图 6（c）中有更多的 AF 聚集到了附近的局部极值点的附近，并有 AF 开始到达了全局极值点附近，从寻优曲线可以看出，AF 能较快地聚集到某个极值点，但可能会陷于其中。

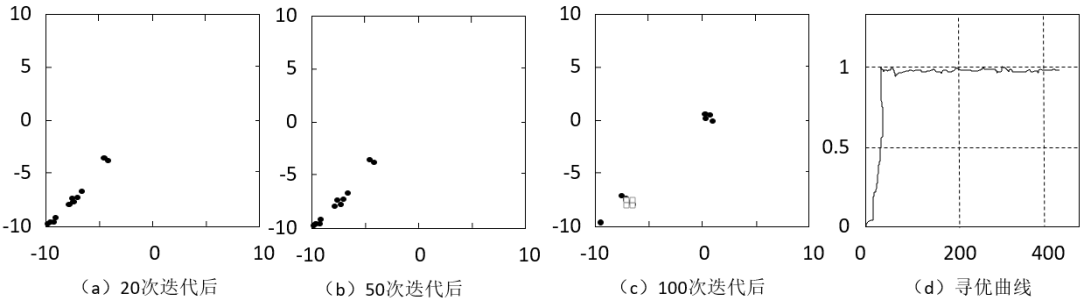


图 6: 追尾行为的仿真结果

图 7 为完整的人工鱼群算法的仿真结果, 从图中可以看出, AF 能迅速向全局极值点附近聚集, 从寻优曲线可以看出, AF 能很快地搜索到全局极值点并较快地稳定在满意解域内。

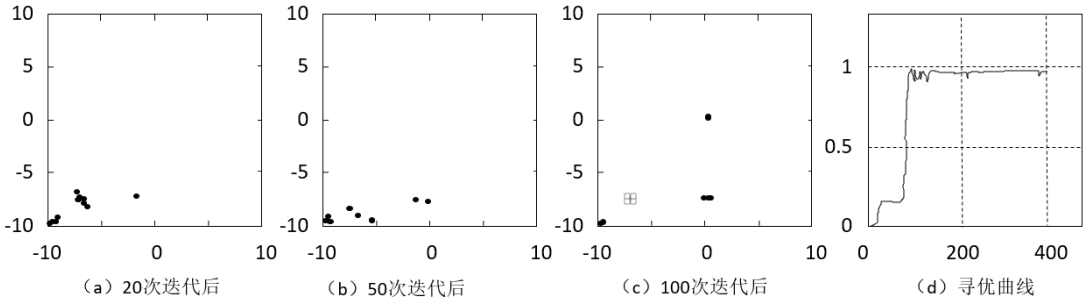


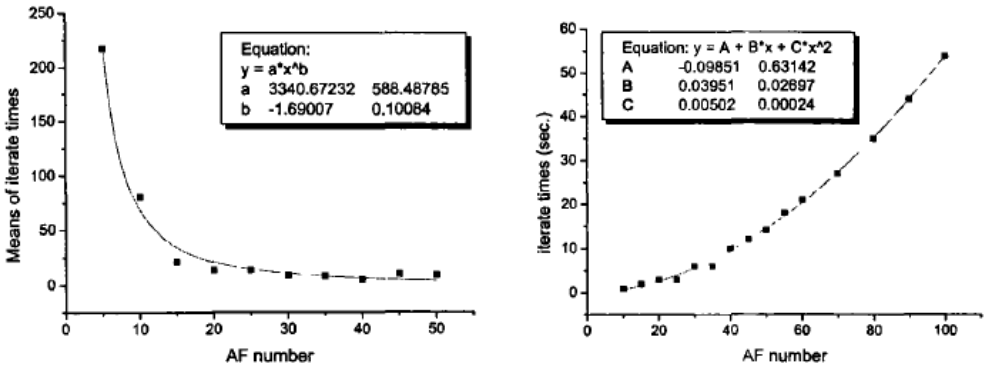
图 7: 鱼群算法仿真结果

由上可得出如下结论：

- 聚群行为能够很好地跳出局部极值, 并尽可能地搜索到其他的极值, 最终搜索到全局极值。
- 追尾行为有助于快速的向某个极值方向前进, 加快寻优的速度, 并防止 AF 在局部振荡而停滞不前。
- 鱼群算法在对以上两种行为进行评价后, 自动选择合适的行为, 从而形成了一种高效快速的寻优策略。

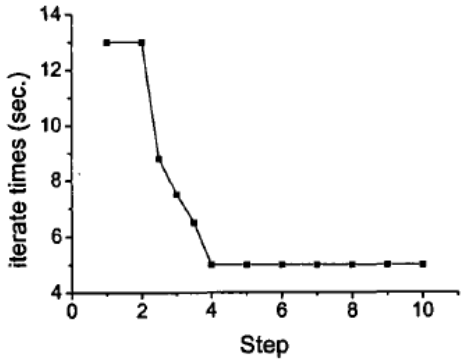
3.2.3 各参数对计算时间的影响

如图所示，可以看出, 参数 Step、Visual、δ 的影响都是有限的, 而人工鱼数目的影响通过回归分析可以看出是呈二次函数上升的; 对人工鱼的数目与收敛迭代次数关系的回归分析可以看出,AFnumher 的增加对迭代次数的减少是呈幂指数下降的, 这或许能补偿一下由于人工鱼数目增加而造成的计算量增加的问题, 因此, 合理选择人工鱼的数目是提高算法效率的关键。这一规律是否具有一般性还有待进一步研究, 如与目标函数的性质有无关系、与算法实现的代码优化有无关系等。



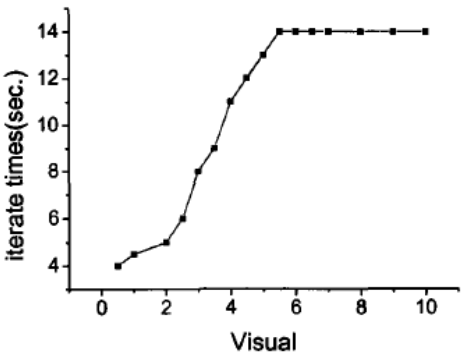
Visual=10, Step=3, delta=0.618

(a) (b)



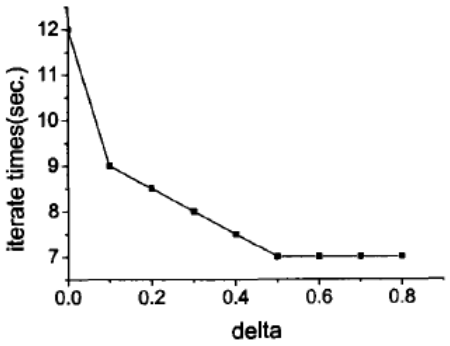
Visual=3, AF number=50, delta=0.618

(c)



AF number=50, Step=3, delta=0.618

(d)



Visual=3, Step=3, AF number=50

(e)

图 8: 各参数对计算时间的影响

3.3 算法对照

遗传算法作为具有全局优化能力的代表算法之一, 受到了长期的、广泛的关注, 作者将鱼群算法与之进行了比较。由于这两种算法都存在一定的随机性, 因此, 在对照中分别进行了 10 组实验, 最后将结果求平均, 横坐标为各算法中迭代的次数。在本实验中, 鱼群算法的参数同上; 遗传算法中, 编码长度为 10, 交叉位数为 4, 变异位数为 2。图 2.7 展示了实验的结果; 为了增强可比性, 在对照 1 中, 鱼群算法的人工鱼个体数为 10, 遗传算法的种群数为 10; 在对照 2 中, 鱼群算法的人工鱼个体数为 50, 遗传算法的种群数为 50。可见, 鱼群算法中当人工鱼个体的数目较少时, 还不能体现出它的优势, 当然, 对于遗传算法来说, 种群数较少时也具有容易陷入局部极值和早熟的可能; 当人工鱼个体的数目增加时, 鱼群算法的收敛速度得以提高, 而遗传算法则由于种群数的增多减缓了进化的速度, 可见鱼群算法中蕴含着集群智能的优势。

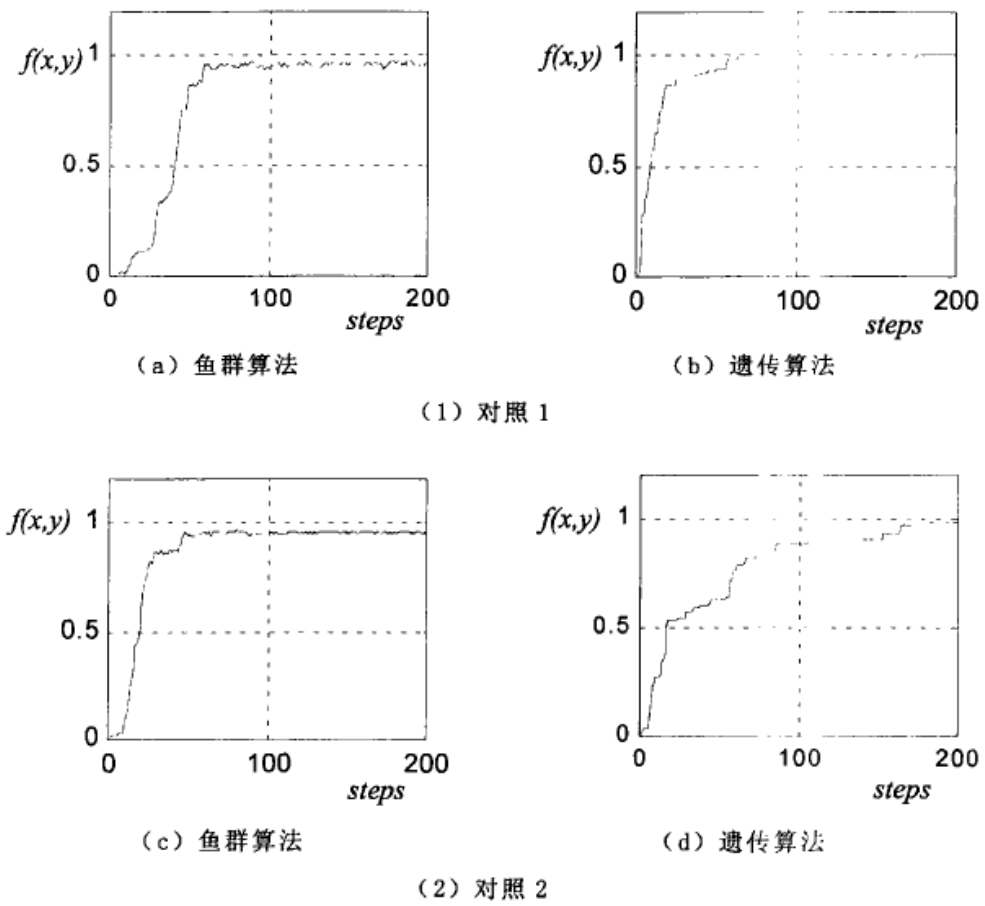


图 9: 鱼群算法与遗传算法的比较

3.4 算法特点

- 算法只需要比较目标函数值，对目标函数的性质要求不高
- 算法对初值的要求不高，初值随机产生或设定为固定值均可以
- 算法对参数设定的要求不高，有较大的容许范围
- 算法具备并行处理的能力，寻优速度较快
- 算法具备全局寻优的能力

4 算法应用

4.1 组合优化问题

4.1.1 算法原理

算法的原理与基本鱼群算法一致, 在描述和实现中有些地方进行了适当的调整。如为了算法的简便易行, 最优值的获取方式采用了跟踪记录最优个体状态的方法, 因此, 引入了公告板 ($AF\_bulletin$) 用来记录最优状加声, 决妒第三章组合优化问题的人工鱼群算法应用 43-态; 为了解决带约束的优化问题, 引入了约束行为 ($AF\_subject$), 扩展了算法的适用范围; 为了更有效的增强算法的效能, 用移动策略 ($AF\_movestrategy$) 替代了行为评价。

约束行为：在寻优过程中, 由于聚群行为、随机行为等操作的作用, 容易使得人工鱼的状态变得不可行, 这时就需要加入相应的约束来对其进行规整化, 使它们由无效状态或不可行状态转变成可行的。
公告板：公告板用来记录最优人工鱼个体的状态。各人工鱼个体在寻优过程中, 每次行动完毕就检验自身状态与公告板的状态, 如果自身状态优于公告板状态, 就将公告板的状态改写为自身状态, 这样就使公告板记录下历史最优的状态。
移动策略：移动策略是原行为评价的一种延伸, 可以依旧采用原行为评价的模式, 也可以采取一定的行动策略, 如先进行追尾行为, 如果没有进步再进行觅食行为, 如果还没有进步则进行聚群行为, 如果依然没有进步就进行随机移动行为。

4.1.2 计算实例

旅行商问题 (TSP,travelingsalesman problem) 是一个典型的组合优化问题, 并且是 NP 一 hard 问题, 下面就以对称距离的 TSP 问题为例来验证算法的性能。
旅行商问题就是一个商人欲到 n 个城市推销商品, 每两个城市之间的距离为 $d_{i,j}$, 如何选择一条道路使得商人每个城市走一遍后回到起点所走过的路径最短。它的数学模型描述为:
$min \sum_{i \neq j} d_{ij} x_{ij}$
$s.t. \sum_{j=1}^{n} x_{ij} = 1, i = 1, 2, ..., n,$
$\sum_{j=1}^{n} x_{ij} = 1, j = 1, 2, ..., n$
$\sum_{i,j \in s} x_{ij} \leq |s| - 1, 2 \leq |s| \leq n - 2, s \subset \{1, 2, ..., n\},$
$x_{ij} \in \{0, 1\}, i, j = 1, 2, ..., n, i \neq j$

数据来源：http://www.uni-heidelberg.de/iwr/comopt/software/TSPLIB95/; 在对城市 Ulysses22（22 个城市）的寻优中，算法参数为 $AF\_Number = 9, Visual = 16, trynumber =$ 500,200 步迭代的计算时间约为 11 秒；在对 Att48（48 个城市）的寻优中，$AF\_Number = 9, Visual = 45, trynumber = 300$, 迭代计算时间大约为 70 秒。
在仿真实验中, 为了反映算法对组合优化问题的解决能力, 没有针对 TSP 问题的特点添加任何的启发规则, 完全依靠人工鱼群算法对解空间的寻优能力, 可见, 人工鱼群算法具有较快的收敛速度, 但是随着问题规模的扩大, 其寻优的精度会有所降低, 这是有待改进的地方。

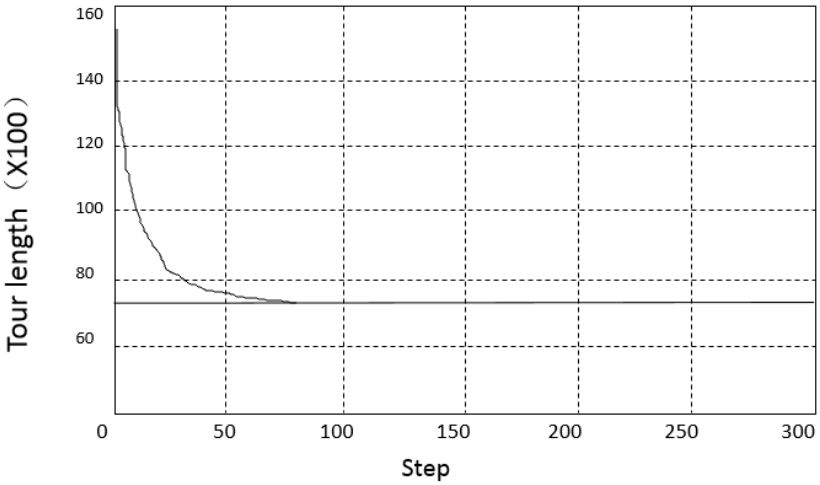


图 10: Ulysses 22 城市 TSP 问题的寻优曲线

在仿真实验中，为了反映算法对组合优化问题的解决能力，没有针对 TSP 问题的特点添加任何的启发规则，完全依靠人工鱼群算法对解空间的寻优能力，可见，人工鱼群算法具

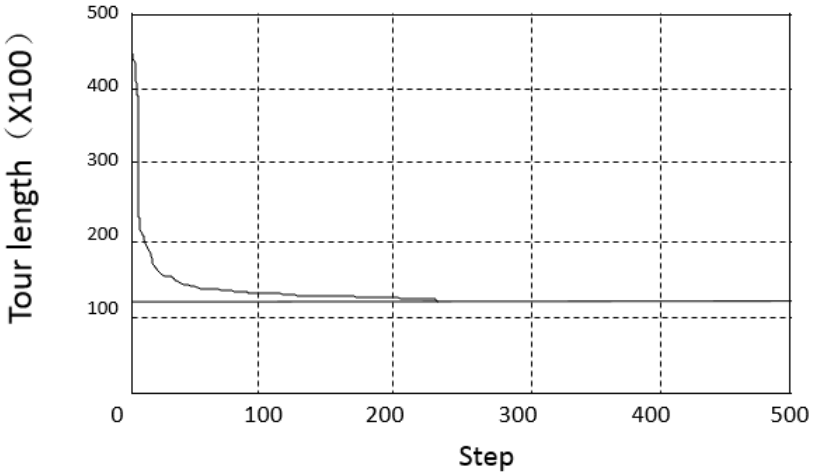


图 11: Att48 城市 TSP 问题的寻优曲线

有较快的收敛速度，但是随着问题规模的扩大，其寻优的精度会有所降低，这是有待改进的地方。

4.1.3 结论

通过对 TSP 的仿真可以看出, 人工鱼群算法在离散问题的解决方面也具备较快的收敛能力。作为一种新型的智能优化方法, 它还有许多需要进一步改进的地方; 其较快的收敛速度, 可以应用于一些有实时性要求的地方; 对于一些精度要求不高的场合, 也可以用它快速的得到一个可行解; 另外, 对问题的了解不必很深, 甚至不需要问题的精确描述, 也使得它的应用范围得以延伸。

4.2 基于分解协调的人工鱼群算法

4.2.1 算法原理

通常, 大系统可以通过变换分解为一些子系统, 一般子系统之间会存在一定的藕合关系, 如果没有祸合关系, 那么问题会变得更加简单了, 对各子系统分别寻优的解将是大系统的最优解。在对大系统进行分解后, 根据各子系统来设计各类人工鱼相应的寻优目标和行为, 从而将鱼群分为多个不同的种类。设想用刃类鱼来分别表示这万个子系统, 然后用万类鱼群通过基本鱼群算法分别求各子系统的最优解。通常情况下, 由于子系统间和约束条件的祸合性, 所得的解都不是所期望的全局最优解。而随着鱼群协调行为的引入, 这个问题得到了很好的解决。

算法采用自下而上的设计方法, 设计的关键就是底层的人工鱼个体行为的实现, 算法的进行也就是人工鱼个体的自适应的行为活动, 最优解将在该过程中突现出来。算法中每条人工鱼作为一个实体, 该实体中封装着自身的状态和行为, 实体每活动一次就是算法的一次迭代。

4.2.2 计算实例

如图所示的换热系统，四股热流的热容流量均为 15000W/°C，流线 III 的冷流体（进口温度 150°C）的热容流量为 13500W/°C，其余两股冷流体的热容流量为 10000W/°C。位于流线 III 上的换热器的传热系数均为 135W/$(m^2 \cdot^\circ C)$, 其余的换热器的传热系数均为 100W/$(m^2 \cdot^\circ C)$. 要求三股冷流体的出口温度分别达到 400°C，450°C，500°C。求换热面积

最小时的换热面积最优分配。

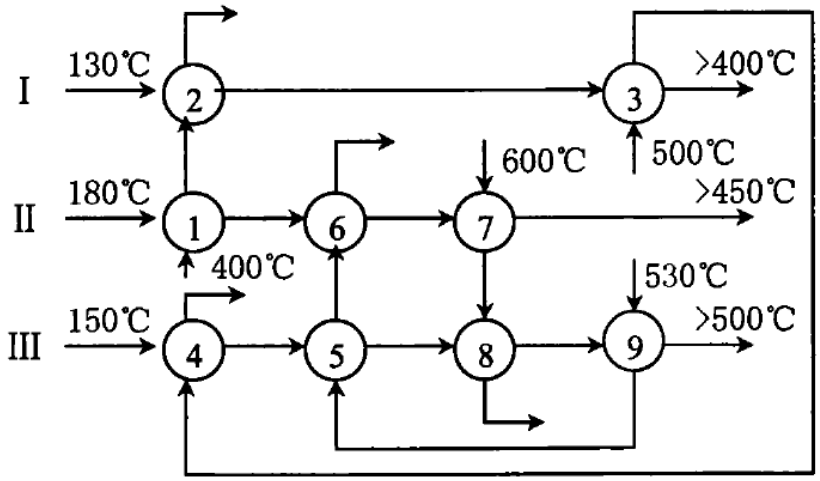


图 12: 换热器系统

计算结果：在实验中每类人工鱼的数目在 10 个左右时就能稳定的收敛到满意的解域, 移动步长 tSeP 的取值从 10 一 200, 视野范围 visual 的取值从 20 一 400 都能保证比较稳定的收敛, 并且迭代 100 次的时间约 5 秒钟。由于算法中存在一定的随机性, 所以为了验证算法的收敛性能, 连续进行了多次计算。参数取值为：每类人工鱼的个数均为 15 个，移动步长 Step=50，视野范围 Visual=300，在仿真中没有使用拥挤度因子 δ。如图所示分别是连续 10 次的计算结果进行平均后所得的迭代收敛曲线以及所得结果最优的一次计算过程的迭代收敛曲线。

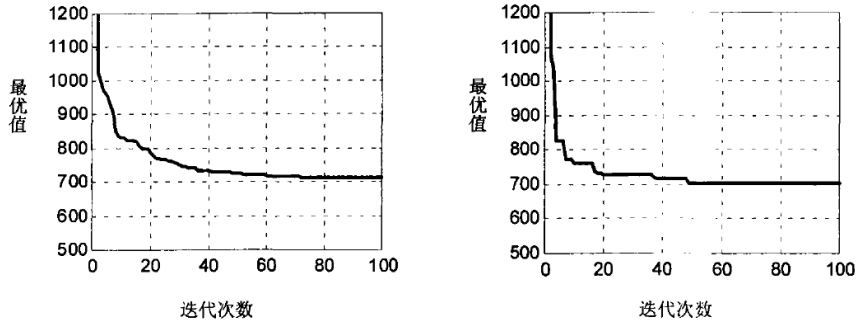


图 13: 计算结果收敛曲线，（左）平均，（右）最优

4.2.3 结论

从以上的计算实例可以看出, 采用基于分解协调思想的人工鱼群算法具有以下几个特点：

- 算法对变量的初始值无要求, 可以是随机产生的值, 包括无效解也可以;
- 算法中仅仅使用了目标函数的函数值, 对函数的性质没有特殊要求;
- 算法的收敛性好, 大都能收敛到比较满意的解;
- 参数的敏感性低, 算法中参数的选择要求不高, 参数的变动对结果的影响也不大;
- 分解与协调思想的引入为复杂的大系统的优化问题的解决提供了一条可行的途径。

5 分析与讨论

5.1 问题

在前面部分的论述中可以看出, 鱼群算法在解决问题时存在一些优点：

- 算法中只需要比较目标函数值, 因此对目标函数要求不高;
- 算法对初值的要求不高, 初值随机产生或设定为固定值均可以;

算法对参数设定的要求不高, 有比较大的容许范围; 同时, 它也存在一些有待改进的地方:

- 随着人工鱼数目的增多, 将会需求更多的存储空间, 也会造成计算量的增长;
- 由于视野和步长的随机性和随机行为的存在, 使得寻优的精度难以很局;
- 由于鱼群模式可以采用面向对象的方式来实现, 所以, 对功能的扩展和改造有着良好的基础。

5.2 改进方法

在以上所述的鱼群算法中, 当寻优的域较大或处于变化平坦的区域时, 一部分 AF 将处于无目的的随机移动中, 这影响了寻优的效率, 下面引入生存机制和竞争机制加以改善。

- 生存机制：当 AF 所处位置的食物能量足以维持其生命时,AF 将按照正常的行为寻优, 否则, 当此处的能量低于其生命的维持所需时, 通常此时处于非全局极值点附近, 寻优通常会没有结果, 所以, 强制该 AF 初始化, 例如, 可以随机产生该 AF 的位置, 使其跳出该区域, 这样就相当于利用相同的存储空间增加了寻优 AF 的个数, 从而提高了算法的效率。
- 竞争机制：随着寻优的逐步进展,AF 的生存周期将被其中最强的竞争者所提升, 从而使得那些处于非全局极值点附近的 AF 能有机会展开更广范围的搜索。

分段优化方法：算法在优化过程初期虽然具有较快的收敛品质, 但在后期却往往收敛较慢, 或者无法达到要求的精度, 因此, 与其他算法相结合, 在合适的时候与它们互相切换, 实现各算法之间的优缺点互补, 也是一种解决问题的常用方法。

混合优化方法: 鱼群模式提供了一种解决问题的架构, 其中自然可以应用传统的一些相对成熟的计算方法, 而面向对象的方法使得将其他计算方法与鱼群算法的有机融合提供了良好的基础。例如, 如果问题的模型比较熟知, 并且目标函数的非线性程度不是非常严重, 则可以在觅食行为中使用单纯行法等传统方法来代替在视野中的随机搜索方法, 这样, 在提高收敛速度的同时, 能适当提高收敛的精度, 并且还能在一定程度上克服局部极值的问题。

5.3 结论

以上可以看出, 人工鱼群算法由于采用了鱼群模式, 使得在扩展性、改造性和融合性等方面具备较好的能力。依靠一种理想的算法来解决所有的问题也是不太符合实际的, 所以, 应用一些措施在一定程度上提高自身算法的品质, 另一方面, 与其他算法相融合, 达到超过单一算法的效能, 这也许才是解决一些难题的有效方法。

参考文献

[1] 李晓磊, 邵之江, 钱积新. 一种基于动物自治体的寻优模式: 鱼群算法 [J]. 系统工程理论与实践,2002,22(11):32 － 38.

[2] 李晓磊. 一种新到的智能优化方法———人工鱼群算法 [D]. 杭州: 浙江大学,2003.

[3] 李晓磊, 钱积新. 基于分解协调的人工鱼群优化算法研究 [J]. 电路与系统学报,2003,8(1) : 1 － 6.