粒子群优化算法介绍

胡晓辉 2002.04

1. 引言

粒子群优化算法(PSO)是一种进化计算技术(evolutionary computation)，有 Eberhart 博士和 kennedy 博士发明。源于对鸟群捕食的行为研究

PSO 同遗传算法类似，是一种基于叠代的优化工具。系统初始化为一组随机解，通过叠代搜寻最优值。但是并没有遗传算法用的交叉(crossover)以及变异(mutation)。而是粒子在解空间追随最优的粒子进行搜索。详细的步骤以后的章节介绍

同遗传算法比较，PSO 的优势在于简单容易实现并且没有许多参数需要调整。目前已广泛应用于函数优化，神经网络训练，模糊系统控制以及其他遗传算法的应用领域

2. 背景: 人工生命

"人工生命"是来研究具有某些生命基本特征的人工系统. 人工生命包括两方面的内容:

1. 研究如何利用计算技术研究生物现象
2. 研究如何利用生物技术研究计算问题

我们现在关注的是第二部分的内容. 现在已经有很多源于生物现象的计算技巧. 例如, 人工神经网络是简化的大脑模型. 遗传算法是模拟基因进化过程的.

现在我们讨论另一种生物系统- 社会系统. 更确切的是, 在由简单个体组成的群落与环境以及个体之间的互动行为. 也可称做"群智能"(swarm intelligence). 这些模拟系统利用局部信息从而可能产生不可预测的群体行为

例如 floys 和 boids, 他们都用来模拟鱼群和鸟群的运动规律, 主要用于计算机视觉和计算机辅助设计.

在计算智能(computational intelligence)领域有两种基于群智能的算法. 蚁群算法(ant colony optimization)和粒子群算法(particle swarm optimization). 前者是对蚂蚁群落食物采集过程的模拟. 已经成功运用在很多离散优化问题上.

粒子群优化算法(PSO) 也是起源对简单社会系统的模拟. 最初设想是模拟鸟群觅食的过程. 但后来发现 PSO 是一种很好的优化工具.

3. 算法介绍

如前所述，PSO 模拟鸟群的捕食行为。设想这样一个场景：一群鸟在随机搜索食物。在这个区域里只有一块食物。所有的鸟都不知道食物在那里。但是他们知道当前的位置离食物还有多远。那么找到食物的最优策略是什么呢。最简单有效的就是搜寻目前离食物最近的鸟的周围区域。

PSO 从这种模型中得到启示并用于解决优化问题。PSO 中，每个优化问题的解都是搜索空间中的一只鸟。我们称之为“粒子”。所有的例子都有一个由被优化的函数决定的适应值(fitness value)，每个粒子还有一个速度决定他们飞翔的方向和距离。然后粒子们就追随当前的最优粒子在解空间中搜索

PSO 初始化为一群随机粒子(随机解)。然后通过叠代找到最优解。在每一次叠代中，粒子通过跟踪两个"极值"来更新自己。第一个就是粒子本身所找到的最优解。这个解叫做个体极值 pBest. 另一个极值是整个种群目前找到的最优解。这个极值是全局极值 gBest。另外也可以不用整个种群而只是用其中一部分最为粒子的邻居，那么在所有邻居中的极值就是局部极值。

在找到这两个最优值时, 粒子根据如下的公式来更新自己的速度和新的位置

$$V_{id} = w \times V_{id} + c_1 \times rand() \times (p_{id} - x_{id}) + c_2 \times Rand() \times (p_{gd} - x_{id}) \quad (1)$$

$$x_{id} = x_{id} + V_{id} \quad (2)$$

v[] 是粒子的速度, persent[] 是当前粒子的位置. pbest[] and gbest[] 如前定义 rand () 是介于（0， 1）之间的随机数. c1, c2 是学习因子. 通常 c1 = c2 = 2.

程序的伪代码如下

```
For each particle {
    Do {
        Initialize particle
}

Do {
    For each particle {
        Calculate fitness value
        If the fitness value is better than the best fitness value (pBest) in history
            set current value as the new pBest
    }
    Choose the particle with the best fitness value of all the particles as the gBest
    For each particle {
        Calculate particle velocity according equation (a)
        Update particle position according equation (b)
    }
} While maximum iterations or minimum criteria is not attained
```

在每一维粒子的速度都会被限制在一个最大速度 Vmax，如果某一维更新后的速度超过用户设定的 Vmax，那么这一维的速度就被限定为 Vmax.

4. 遗传算法和 PSO 的比较

大多数演化计算技术都是用同样的过程
1. 种群随机初始化
2. 对种群内的每一个个体计算适应值(fitness value).适应值与最优解的距离直接有关
3. 种群根据适应值进行复制
4. 如果终止条件满足的话，就停止，否则转步骤 2

从以上步骤，我们可以看到 PSO 和 GA 有很多共同之处。两者都随机初始化种群，而且都使用适应值来评价系统，而且都根据适应值来进行一定的随机搜索。两个系统都不是保证一定找到最优解

但是，PSO 没有遗传操作如交叉(crossover)和变异(mutation). 而是根据自己的速度来决定搜索。粒子还有一个重要的特点，就是有记忆。

与遗传算法比较, PSO 的信息共享机制是很不同的. 在遗传算法中，染色体(chromosomes) 互相共享信息，所以整个种群的移动是比较均匀的向最优区域移动. 在 PSO 中, 只有 gBest (or lBest) 给出信息给其他的粒子， 这是单向的信息流动. 整个搜索更新过程是跟随当前最优解的过程. 与遗传算法比较, 在大多数的情况下，所有的粒子可能更快的收敛于最优解

5. 人工神经网络 和 PSO

人工神经网络(ANN)是模拟大脑分析过程的简单数学模型，反向转播算法是最流行的神经网络训练算法。进来也有很多研究开始利用演化计算(evolutionary computation)技术来研究人工神经网络的各个方面。

演化计算可以用来研究神经网络的三个方面：网络连接权重，网络结构(网络拓扑结构，传递函数)，网络学习算法。

不过大多数这方面的工作都集中在网络连接权重，和网络拓扑结构上。在 GA 中，网络权重和/或拓扑结构一般编码为染色体(Chromosome)，适应函数(fitness function)的选择一般根据研究目的确定。例如在分类问题中，错误分类的比率可以用来作为适应值

演化计算的优势在于可以处理一些传统方法不能处理的例子例如不可导的节点传递函数或者没有梯度信息存在。但是缺点在于：在某些问题上性能并不是特别好。2. 网络权重的编码而且遗传算子的选择有时比较麻烦

最近已经有一些利用 PSO 来代替反向传播算法来训练神经网络的论文。研究表明 PSO 是一种很有潜力的神经网络算法。PSO 速度比较快而且可以得到比较好的结果。而且还没有遗传算法碰到的问题

这里用一个简单的例子说明 PSO 训练神经网络的过程。这个例子使用分类问题的基准函数(Benchmark function)IRIS 数据集。(Iris 是一种鸢尾属植物) 在数据记录中，每组数据包含 Iris 花的四种属性：萼片长度，萼片宽度，花瓣长度，和花瓣宽度，三种不同的花各有 50 组数据. 这样总共有 150 组数据或模式。

我们用 3 层的神经网络来做分类。现在有四个输入和三个输出。所以神经网络的输入层有 4 个节点，输出层有 3 个节点我们也可以动态调节隐含层节点的数目，不过这里我们假定隐含层有 6 个节点。我们也可以训练神经网络中其他的参数。不过这里我们只是来确定网络权重。粒子就表示神经网络的一组权重，应该是 4*6+6*3=42 个参数。权重的范围设定为[-100，100] (这只是一个例子，在实际情况中可能需要试验调整).在完成编码以后，我们需要确定适应函数。对于分类问题，我们把所有的数据送入神经网络，网络的权重有粒子的参数决定。然后记录所有的错误分类的数目作为那个粒子的适应值。现在我们就利用 PSO 来训练神经网络来获得尽可能低的错误分类数目。PSO 本身并没有很多的参数需要调整。所以在实验中只需要调整隐含层的节点数目和权重的范围以取得较好的分类效果。

6. PSO 的参数设置

从上面的例子我们可以看到应用 PSO 解决优化问题的过程中有两个重要的步骤: 问题解的编码和适应度函数
PSO 的一个优势就是采用实数编码, 不需要像遗传算法一样是二进制编码(或者采用针对实数的遗传操作.例如对于问题 f(x) = x1^2 + x2^2+x3^2 求解, 粒子可以直接编码为 (x1, x2, x3), 而适应度函数就是 f(x). 接着我们就可以利用前面的过程去寻优.这个寻优过程是一个叠代过程, 中止条件一般为设置为达到最大循环数或者最小错误

PSO 中并没有许多需要调节的参数,下面列出了这些参数以及经验设置

粒子数: 一般取 20 – 40. 其实对于大部分的问题 10 个粒子已经足够可以取得好的结果, 不过对于比较难的问题或者特定类别的问题, 粒子数可以取到 100 或 200

粒子的长度: 这是由优化问题决定, 就是问题解的长度

粒子的范围: 由优化问题决定,每一维可是设定不同的范围

Vmax: 最大速度,决定粒子在一个循环中最大的移动距离,通常设定为粒子的范围宽度,例如上面的例子里,粒子 (x1, x2, x3) x1 属于 [-10, 10], 那么 Vmax 的大小就是 20

学习因子: c1 和 c2 通常等于 2. 不过在文献中也有其他的取值. 但是一般 c1 等于 c2 并且范围在 0 和 4 之间

中止条件: 最大循环数以及最小错误要求. 例如, 在上面的神经网络训练例子中, 最小错误可以设定为 1 个错误分类, 最大循环设定为 2000, 这个中止条件由具体的问题确定.

全局 PSO 和局部 PSO: 我们介绍了两种版本的粒子群优化算法: 全局版和局部版. 前者速度快不过有时会陷入局部最优. 后者收敛速度慢一点不过很难陷入局部最优. 在实际应用中, 可以先用全局 PSO 找到大致的结果,再有局部 PSO 进行搜索.

另外的一个参数是惯性权重, 由 Shi 和 Eberhart 提出, 有兴趣的可以参考他们 1998 年的论文(题目: A modified particle swarm optimizer)

7.PSO 网上资源

粒子群优化算法的研究还处于初期阶段, 还有很多未知的领域需要研究, 例如关于粒子群理论的数学证明
不过网上已经由了很多的关于粒子群的资源, 下面列出一些:

http://www.particleswarm.net 关于粒子群理论的各方面资源

http://icdweb.cc.purdue.edu/~hux/PSO.shtml 有一份比较全的文献列表以及网上论文

http://www.researchindex.com/ 可以搜索到关于 PSO 的很多论文及文献

参考文献

http://www.engr.iupui.edu/~eberhart/

http://users.erols.com/cathyk/jimk.html

http://www.alife.org

http://www.aridolan.com

http://www.red3d.com/cwr/boids/

http://iridia.ulb.ac.be/~mdorigo/ACO/ACO.html

http://www.engr.iupui.edu/~shi/Coference/psopap4.html

Kennedy, J. and Eberhart, R. C. Particle swarm optimization. Proc. IEEE int'l conf. on neural networks Vol. IV, pp. 1942-1948. IEEE service center, Piscataway, NJ, 1995.

Eberhart, R. C. and Kennedy, J. A new optimizer using particle swarm theory. Proceedings of the sixth international symposium on micro machine and human science pp. 39-43. IEEE service center, Piscataway, NJ, Nagoya, Japan, 1995.

Eberhart, R. C. and Shi, Y. Particle swarm optimization: developments, applications and resources. Proc. congress on evolutionary computation 2001 IEEE service center, Piscataway, NJ., Seoul, Korea., 2001.

Eberhart, R. C. and Shi, Y. Evolving artificial neural networks. Proc. 1998 Int'l Conf. on neural networks and brain pp. PL5-PL13. Beijing, P. R. China, 1998.

Eberhart, R. C. and Shi, Y. Comparison between genetic algorithms and particle swarm optimization. Evolutionary programming vii: proc. 7th ann. conf. on evolutionary conf., Springer-Verlag, Berlin, San Diego, CA., 1998.

Shi, Y. and Eberhart, R. C. Parameter selection in particle swarm optimization. Evolutionary Programming VII: Proc. EP 98 pp. 591-600. Springer-Verlag, New York, 1998.

Shi, Y. and Eberhart, R. C. A modified particle swarm optimizer. Proceedings of the IEEE International Conference on Evolutionary Computation pp. 69-73. IEEE Press, Piscataway, NJ, 1998