遗传算法基础实践报告

——遗传算法对神经网络参数初始化==相关研究与度约束的最小生成树问题实验与分析

学号	姓名	学院	专业
1120192305	孙华山	计算机学院	人工智能

一.实验简介

- 本实验主要基于遗传算法和神经网络结合，进行遗传算法对神经网络进行参数初始化的相关实验与分析；同时通过遗传算法，求解经典组合问题——带约束的最小生成树问题；通过设计算法、代码实践进一步理解掌握遗传算法的基本原理，同时通过实验结果分析，观察遗传算法的相关特点和应用场景等

二.实验目标

1. 通过阅读相关文献，了解遗传算法等随机搜索算法在神经网络相关工作领域的应用
2. 通过阅读文献，了解遗传算法在求解组合问题中的应用，同时学习求解带约束的最小生成树问题
3. 通过代码实践遗传算法，进一步在实际应用中掌握理解算法的原理，体会各个步骤
4. 通过代码实践实现基于遗传算法的神经网络参数初始化优化问题；并进一步进行实验结果的分析，观察遗传算法的特点以及在神经网络应用的优缺点。
5. 通过代码实践，解决带约束的最小生成树问题，与神经网络实验结合分析遗传算法特点以及在组合优化问题中的优缺点
6. 进一步通过实验验证自己的想法，或者分析结果，得出更多结论

三.任务情景描述

1.基于前馈神经网络的分类问题

- 数据描述：

类别	C_1			C_2			C_3		
样本	x_1	x_2	x_3	x_1	x_2	x_3	x_1	x_2	x_3
1	1.58	2.32	-5.8	0.21	0.03	-2.21	-1.54	1.17	0.64
2	0.67	1.58	-4.78	0.37	0.28	-1.8	5.41	3.45	-1.33
3	1.04	1.01	-3.63	0.18	1.22	0.16	1.55	0.99	2.69
4	-1.49	2.18	-1.39	-0.24	0.93	-1.01	1.86	3.19	1.51
5	-0.41	1.21	-4.37	-1.18	0.39	-0.39	1.68	1.79	-0.87
6	1.39	3.16	2.87	0.74	0.96	-1.16	3.51	-0.22	-1.39
7	1.20	1.40	-1.89	-0.38	1.94	-0.48	1.40	-0.44	0.92
8	-0.92	1.44	-3.22	0.02	0.72	-0.17	0.44	0.83	1.97
9	0.45	1.33	-4.38	0.44	1.31	-0.14	0.25	0.68	-0.99
10	-0.76	0.84	-1.96	0.46	1.49	0.68	-0.66	-0.45	0.08

- 总共30个样本，每个样本输入特征数为，一共三个类别；
- 令训练集和测试集的比例为7：3；在实验过程中，在每个类别中随机选取3个样本作为测试集

2.带约束的最小生成树问题

- 问题描述：对n个节点，要求找到n-1条边使得生成的树的权值之和最小
- 各节点权值如下：

```
[[  0 224 224 361 671 300 539 800 943]
 [  0   0 200 200 447 283 400 728 762]
 [  0   0   0 400 566 447 600 922 949]
 [  0   0   0   0 400 200 200 539 583]
 [  0   0   0   0   0 600 447 781 510]
 [  0   0   0   0   0   0 283 500 707]
 [  0   0   0   0   0   0   0 361 424]
 [  0   0   0   0   0   0   0   0 500]
 [  0   0   0   0   0   0   0   0   0]]
```

- 度约束为3

四.GA-NN算法设计

1.遗传算法设计

(1).遗传算法流程

1. 随机产生初始种群
2. 计算群体中的每个个体的适应度，记录最优解
3. 遗传进化：
    a. 选择父类，父类数目与旧群体数量相当（可重复）
    b. 交叉遗传：随机选择父类中的两两组合发生交叉，产生子代数量与父代相当
    c. 突变遗传：根据概率突变，对每个个体进行突变
4. 判断是否达到终止条件，是则退出；反之，返回步骤2
5. 选择记录中最优解

(2).相关设计

1. 编码方式：
    a. 二进制编码
    b. 实数编码：（更好？）
        - 神经网络一共52个参数，因此直接使用52位长的编码表示每一个个体；每一位是一个实数，表示相应的参数
        - 前28位为第一层线性层参数；后24为第二层线性层参数
2. 父类选择
    a. 方法一：轮盘赌法
    b. 方法二：将种群从差到好进行排序，平均分为3部分，分别以概率$0.6, 0.8, 1$的概率进行选择
    - 选择出与原始种群数目相当的群体，可以重复
3. 遗传算子
    - 使用遗传算子时，按照各部分与各部分相互交叉组合的方法进行
    a. 交叉遗传
        - 由于是实数编码，因此进行仿二进制基因交叉：

          每个维度分别计算：

$$o_1 = \overline{x} - \frac{1}{2}\beta(p_2 - p_1)$$

$$o_2 = \overline{x} + \frac{1}{2}\beta(p_2 - p_1)$$

$$where \quad \overline{x} = \frac{1}{2}(p_2 + p_1) \quad p_2 > p_1$$

参数β的选择方法：

$$p(\beta_i) = \begin{cases} 0.5(\eta_c + 1)\beta_i^{\eta_c}, & \text{if } \beta_i \leq 1 \\ 0.5(\eta_c + 1)\frac{1}{\beta_i^{\eta_c + 2}}, & \text{otherwise.} \end{cases}$$

这样使得β尽量在$[1, 2]$，η_c是超参数，设为2

b. 突变遗传

- 遍历每一位，以概率进行如下正态突变：

$$x_{next} = x_{now} + N(0, \sigma)$$

4. 适应度函数

- 由于是多分类问题，使用**F1**范数最为适应度函数，数值越大，表示模型越好，适应度越好

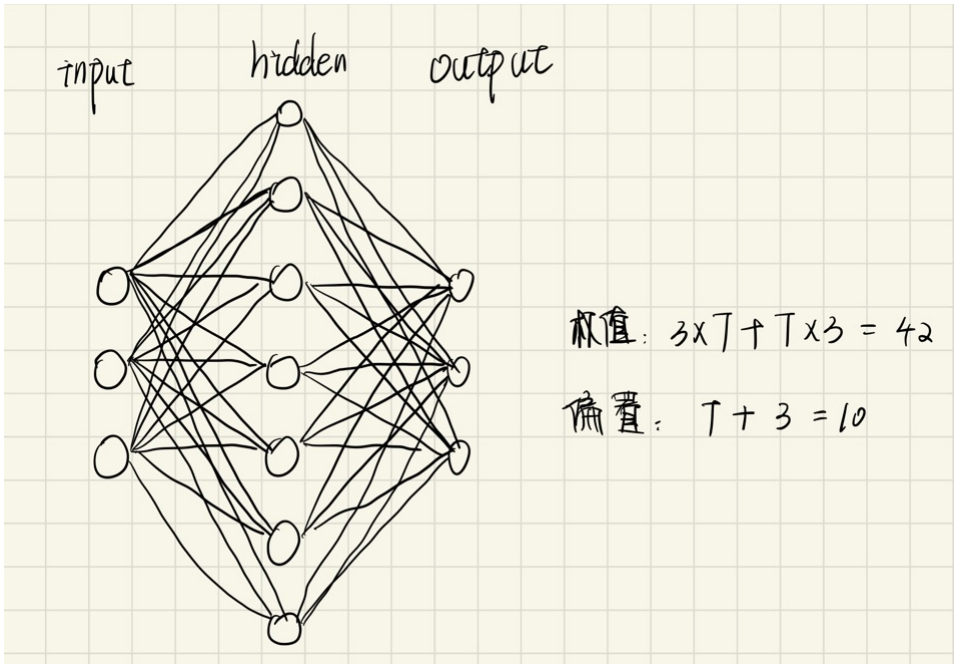
5. 其他参数设置

2.神经网络设计

(1).相关设计

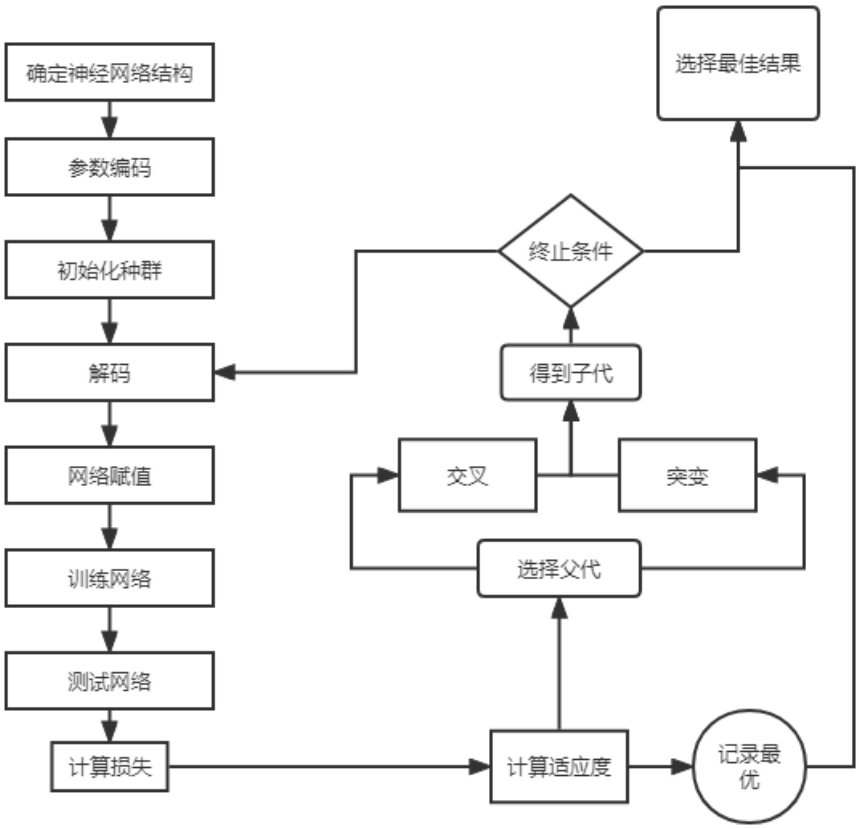
- 使用三层前馈全连接神经网络
  - 输入层：3个神经元
  - 隐藏层：7个神经元
  - 输出层：3个神经元
- 优化算法：
  - AdaGrad优化器
- 损失函数：
  - 交叉熵损失

(2).网络结构

- 

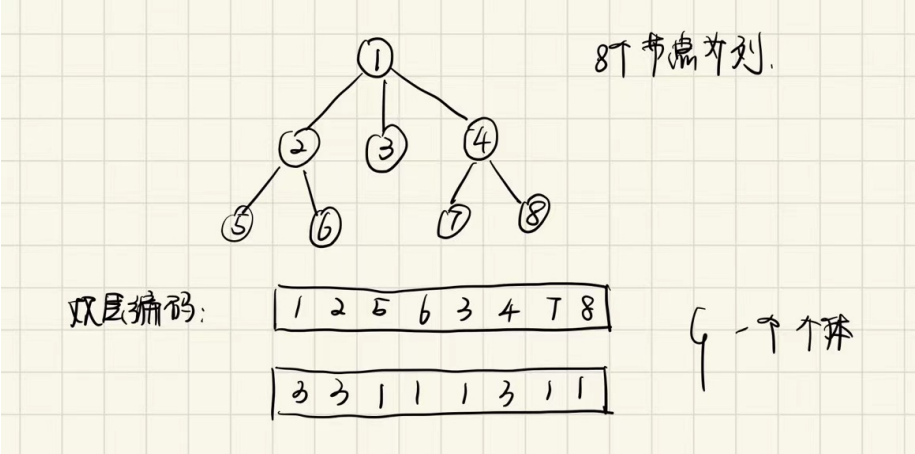
3.实验算法设计

算法流程图如下：



五.GA-MST算法设计

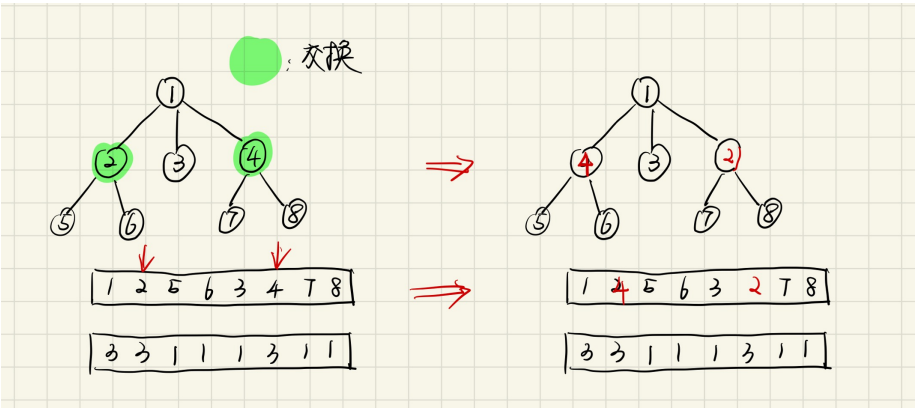
遗传算法相关设计

1. 编码设计：
    - 要综合考虑1.顶点之间的连接关系，2.每个顶点的度的关系。
    - 编码如下（基于PrimPred方法编码）
        - 使用两层，一层为顶点层，取值为$[1,n]$不可重复数；一层为度层，数值为对应点的度，取值为$[1,d],d$为约束度
    - 编码解码过程:
        - 点顶层的排列顺序实际就是一个从根节点从左向右的深度优先遍历的过程
        - 

2. 种群初始化：
    - 初始化种群的基因需要满足：
        i. 任意顶点的度不小于1，所有顶点度之和为$2(n-1)$
        ii. 当前节点与余下顶顶点的度之和不小于d_{rest}（未在染色体度层印指定度值的顶点的度值之和)，也不大于$2(n-1)-d_{used}$（已经在染色体度层中指定度值的顶点度值之和)
3. 父类选择：
    - 同神经网络实验，使用比例法或者轮盘赌法
4. 遗传算子：
    a. 交叉算子：
        - 使用顶点的次序交叉：

b. 突变算子：

- 突变算子决定了局部搜索能力，此问题中使用交换变异：



5. 适应度函数：

- MST的权值之和要求最小，直接使用其倒数作为适应度函数即可：

$$Fitness = -W(T)$$

六.实验环境

本实验在本人的笔记本电脑下进行，计算机配置和实验环境如下：

1. 实验平台：

   Visual Studio Code

   版本: 1.60.2 (user setup)

2. 运行环境：

操作系统	**WINDOWS 10**操作系统
处理器	Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz 2.60 GHz
显卡	Intel(R) UHD Graphics 630; NVIDIA GeForce GTX 16550;
语言	python
依赖包	Numpy、Paddle、matplotlib

七.实验过程

1.神经网络部分

1. 首先使用神经网络直接进行数据的拟合
2. 根据设计的算法，手写遗传算法
3. 多次运行，得到结果

2.度约束的最小生成树问题

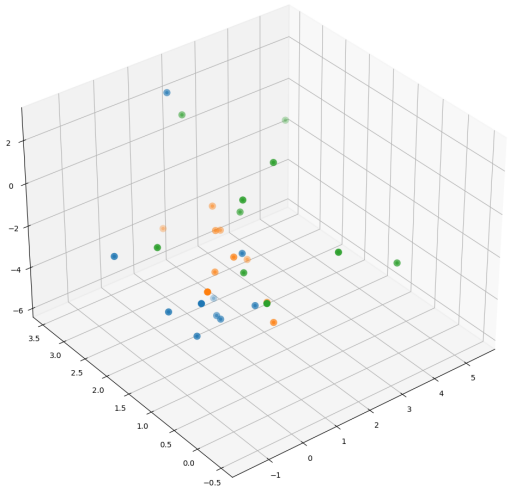
1. 根据设计的算法，手写了遗传算法，基于第一部分的代码，进行部分修改得到
2. 在不同参数下，多次运行，得到结果

八.实验结果与分析

1.神经网络部分

1. 实验结果

    a. 数据可视化：



b. 网络summary:

```
 Layer (type)      Input Shape       Output Shape       Param #
================================================================
   Linear-1       [[1, 1, 3]]        [1, 1, 7]           28
    ReLU-1        [[1, 1, 7]]        [1, 1, 7]            0
   Linear-2       [[1, 1, 7]]        [1, 1, 3]           24
================================================================
Total params: 52
Trainable params: 52
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 0.00
Forward/backward pass size (MB): 0.00
Params size (MB): 0.00
Estimated Total Size (MB): 0.00
----------------------------------------------------------------


{'total_params': 52, 'trainable_params': 52}
```

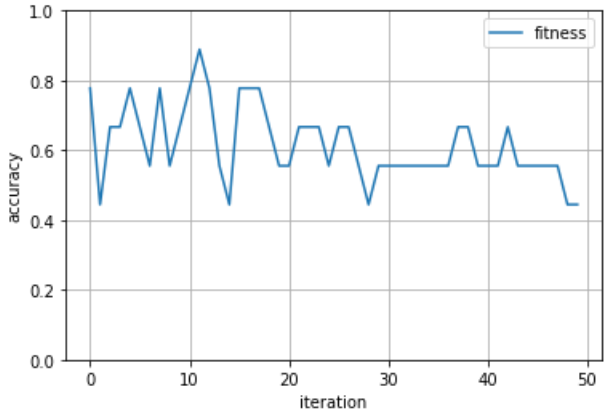
c. 在以下参数下学习得到模型，

最优个体参数：在测试集上准确率为100%

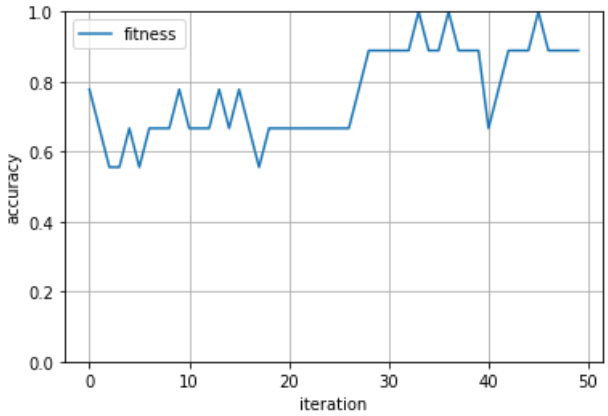
```
第47代，最佳适应度为0.8888888888888888
第48代，最佳适应度为0.8888888888888888
第49代，最佳适应度为0.8888888888888888
the best one is 1.0 [0.8669871867657424, -16.306491727543136, 5.865154163604679, -96.30662314959505, 27.651699749854256, 66.45360718593696, 5.389958518425463, 2.7115953441497807, -16.696839703712044, -8.767615138944958, 0.45609766623403747, 16.599162907760835, 7.748554627144617, 18.626782830670 59, 3.545142028277335, 10.29057442115628, 17.483040256871547, -46.06065001295052, 8.22233379482829, -6.06892409543897, -15.30930397237702, 4.984772678255586, 6.135528420298662, 24.785050123570258, 11.721045458274759, 33.562452297894545, 51.38197007115053, -45.57132787309578, -33.38660617601376, 6.052750879920863, -20.53131207327172, 8.11102121033726, 12.802676195547033, 37.36107572302746, -35.28478544047801, -21.697880896593386, -26.480151862912127, 1.409594521902179, -7.933214141463052, -0.33590104334791837, -33.42300404647801, 6.6986473217518485, 2.042464046854398, -16.361887911384123, 5.182952658142614, -6.009962562324771, -5.284265303264669, -9.51254267365227, -16.961157393751787, -9.204814725043544, 17.49590620858733, -12.848176023156347]
```

各代最优个体的准确率变化：

- 一次实验



- 二次实验：



2. 分析

a. 通过第一次实验在测试集上的准确率图像的变化我们观察到，在遗传算法迭代过程中，起初随着迭代，训练效果是比价好的，后来反而有所下降；而在第二次实验中，遗传算法表现非常出色，总体趋势是逐渐上升，最终基本收敛到最优值；分析认为这是由于编码不能很好体现网络信息，未能充分利用其启发式信息进行搜索的原因，在这种情况下，遗传算法就相当于随即搜索；通过两次实验，可以知道遗传算法具有一定的效果，但是由于随机性，搜索方向是无法确定的，搜索到最优解的时机也是无法预见的。

b. 由于数据量较小，因此能够看到最终的测试效果在不同初始化情况下是有一定差距的，因此模型训练效果子在较小数据上是与参数的初始化有一定关联。这也是为什么会出现第二次实验结果的原因。因此，在较小数据集上，通过遗传算法进行网络参数初始化是具有一定作用的。

c. 但是在实验过程中，由于遗传算法是分布式大规模搜索，故在实际网络优化过程中，其开销肯定是较大的；并且，通过查询资料发现：在数据量非常大的情况下，模型最终的训练效果和初始化的参数关联程度不大，使用遗传算法在大数据情况下是不必要的，直接使用用SGD、Adam等算法进行模型优化在实际过程会是更好的选择。

d. 总结认为：

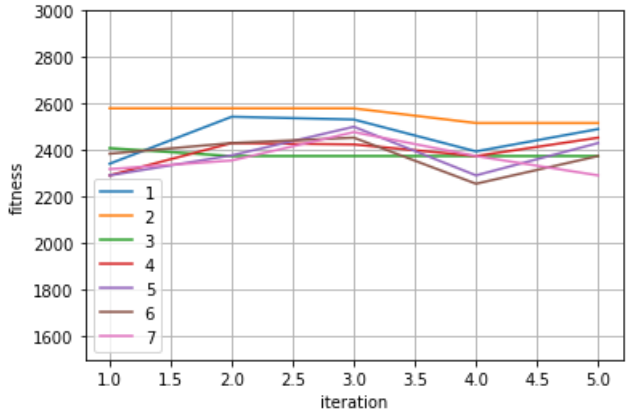
- 神经网络参数编码不一定能够讲问题信息很好嵌入基因中，因此搜索的随机性较大；
- 其次，当数据量较大时，参数随机初始化，均能够基本收敛到较好的相同的效果，因此，在优化神经网络的过程中，重点在参数设置、训练算法等过程，而不是初始化参数；
- 同时，神经网络训练需要一定开销，加上使用遗传算法大规模搜索，开销更大；
- 因此在实际工程中不是最好的选择

2.度约束的最小生成树问题

1. 实验结果：

种群规模	迭代次数	突变概率	最优值	基因（深搜解码）	运行**5**次结果
100	200	0.2	2342	[[1. 2. 3. 6. 4. 8. 7. 5. 0.] [3. 1. 3. 3. 2. 1. 1. 1. 1.]]	2342,2544,2532,2395,2491
50	200	0.2	2517	[[7. 5. 3. 6. 4. 8. 1. 2. 0.] [1. 3. 3. 1. 2. 1. 3. 1. 1.]]	2580,2580,2580,2517,2517
100	500	0.2	2375	[[1. 5. 6. 7. 8. 3. 4. 2. 0.] [3. 3. 3. 1. 1. 2. 1. 1. 1.]]	2409,2375,2375,2375,2375
100	200	0.01	2292	[[2. 1. 5. 3. 6. 8. 7. 4. 0.] [2. 2. 2. 3. 3. 1. 1. 1. 1.]]	2292,2431,2425,2375,2454
100	200	0.4	2292	[[2. 1. 3. 4. 5. 6. 7. 8. 0.] [1. 3. 3. 1. 2. 3. 1. 1. 1.]]	2292,2378,2501,2292,2431
100	500	0.4	2256	[[2. 1. 4. 3. 6. 8. 7. 5. 0.] [2. 3. 1. 3. 3. 1. 1. 1. 1.]]	2385,2431,2454,2256,2375
100	500	0.7	2292	[[2. 1. 5. 3. 6. 7. 8. 4. 0.] [2. 3. 1. 3. 3. 1. 1. 1. 1.]]	2319,2356,2478,2375,2292

- 以上结果绘图：

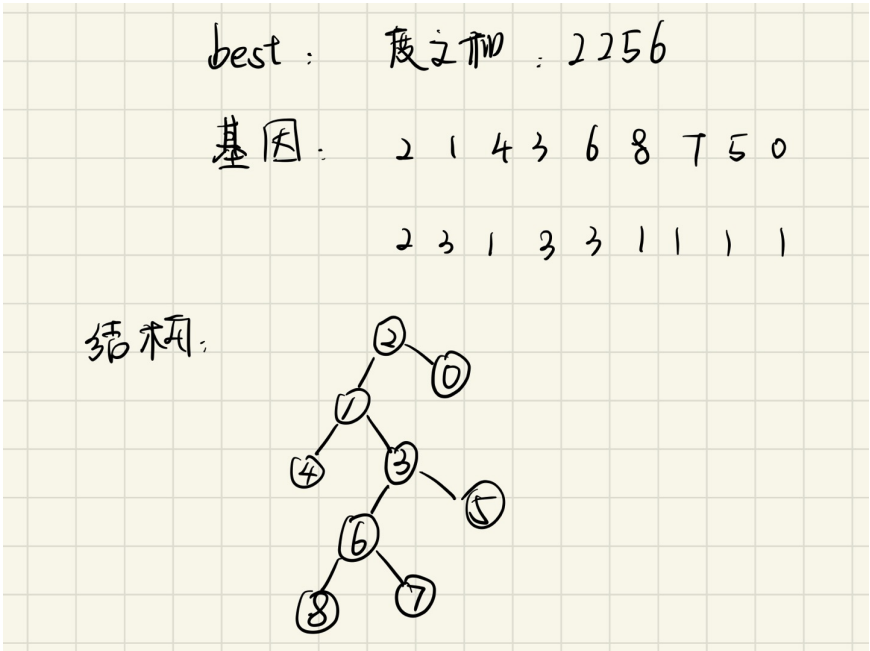


- 最佳度约束的最小生成树为：

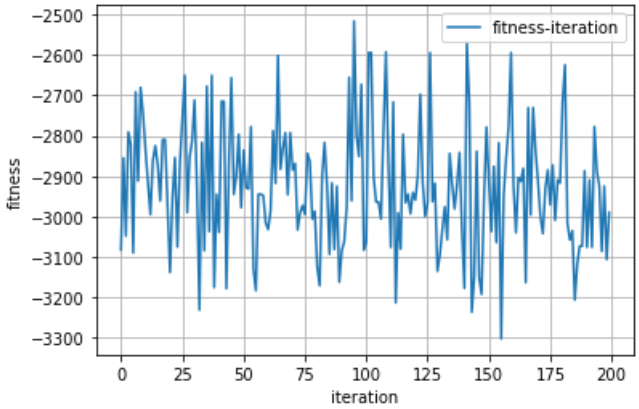
  实验结果：

```
the best one is -2385 [[1. 3. 6. 7. 8. 4. 2. 0. 5.]
 [2. 3. 2. 2. 1. 1. 2. 2. 1.]]
权重之和为： 2385
[(1.0, 3.0), (3.0, 6.0), (6.0, 7.0), (7.0, 8.0), (3.0, 4.0), (1.0, 2.0), (2.0, 0.0), (0.0, 5.0)]
the best one is -2431 [[2. 0. 1. 3. 6. 5. 7. 8. 4.]
 [2. 1. 2. 3. 3. 2. 1. 1. 1.]]
权重之和为： 2431
[(2.0, 0.0), (2.0, 1.0), (1.0, 3.0), (3.0, 6.0), (6.0, 5.0), (5.0, 7.0), (6.0, 8.0), (3.0, 4.0)]
the best one is -2454 [[5. 1. 2. 3. 4. 8. 6. 7. 0.]
 [3. 2. 1. 3. 2. 1. 2. 1. 1.]]
权重之和为： 2454
[(5.0, 1.0), (1.0, 2.0), (5.0, 3.0), (3.0, 4.0), (4.0, 8.0), (3.0, 6.0), (6.0, 7.0), (5.0, 0.0)]
the best one is -2256 [[2. 1. 4. 3. 6. 8. 7. 5. 0.]
 [2. 3. 1. 3. 3. 1. 1. 1. 1.]]
权重之和为： 2256
[(2.0, 1.0), (1.0, 4.0), (1.0, 3.0), (3.0, 6.0), (6.0, 8.0), (6.0, 7.0), (3.0, 5.0), (2.0, 0.0)]
the best one is -2375 [[7. 6. 5. 3. 4. 1. 2. 0. 8.]
 [1. 3. 3. 2. 1. 3. 1. 1. 1.]]
权重之和为： 2375
[(7.0, 6.0), (6.0, 5.0), (5.0, 3.0), (3.0, 4.0), (5.0, 1.0), (1.0, 2.0), (1.0, 0.0), (6.0, 8.0)]
```

  结构：



- 某次实验各代最佳适应度与迭代次数关系：



2. 分析：

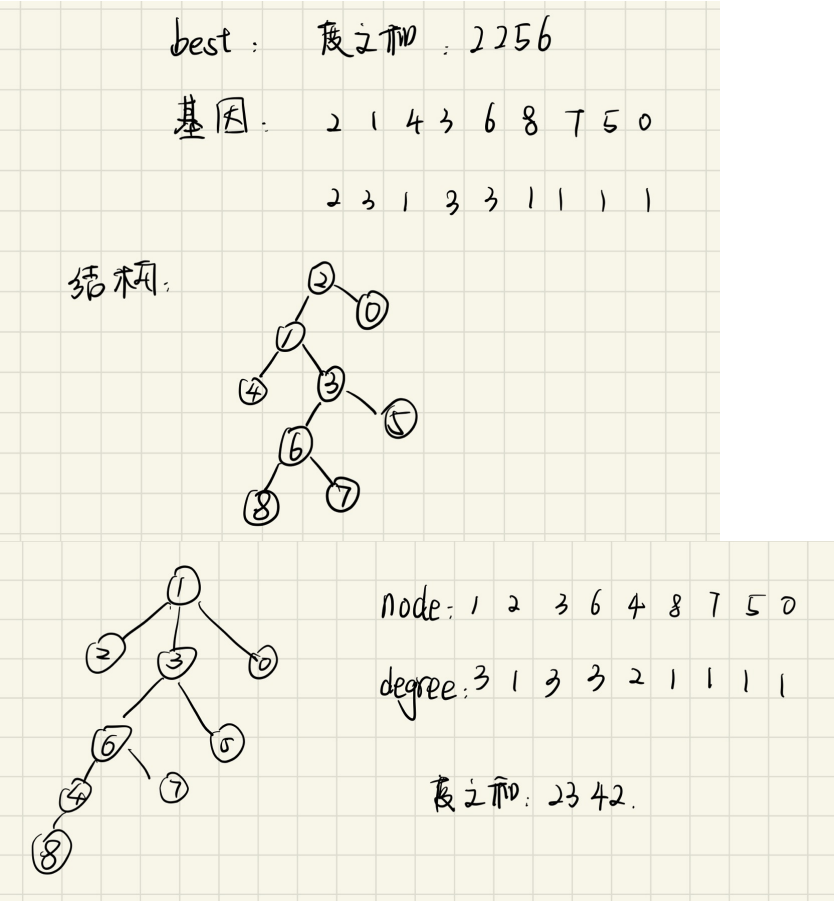
a. 首先，通过在不同实验设置下，进行运行的结果以及绘制的图可以看出，结果基本是相近的，这说明遗传算法具有较好的鲁棒性，相同参数设置下均能搜索到最优值附近。

b. 其次，通过实验结果发现，在种群规模较大（1，2组）、迭代次数较大（5，6组）、突变概率较大（1，5组）的情况下，算法搜索能力较强，所得到的最佳结果更加优秀；同时，我们在其他组能发现，这种规律不是绝对的，如1，4组，6，7组；因此：种群规模越大、迭代次数较大、突变概率较大，搜索能力越强的规律是大致趋势，需要平衡好“探索与利用”：种群规模和迭代次数适当大，增强全局搜索能力；同时突变概率适当，不应过大，不仅是自然规律，同时也是为了使得迭代初期主要在全局搜索；后期局部精细搜索，若太大，会导致突变带来振荡。

c. 通过最佳适应度和迭代关系图可以发现，遗传算法振荡是非常强的，前期适应度逐渐变高，但是后又变化较大，贬低；最佳的个体不一定在最后形成；因此，遗传算法是随机大规模的搜索算法，其具有一定的随机性，收敛性不能保证，振荡较大。

d. 我认为，遗传算法之所以能搜索到较优秀的结果的原因一定程度上是其随机性和搜索规模较大的原因，同样也是由于其随机性，为搜索方向、收敛带来一定问题。

e. 同时虽然各个值有差异，但是通过解码发现，实际上树结构是相同的，差异在于节点的连接，因此通过GA，搜索到了较好的树结构：



两个不同结果的**MST**结构图

3.对比分析

通过对比遗传算法在神经网络和度约束的MST问题中的结果；我认为：

1. 遗传算法更加适用于组合优化问题：
    a. 一定程度上在于这中离散的分配问题可以较好地进行编码，在基因中嵌入更多的问题信息以启发搜索，同时由于基因更加能够关联实际问题的结构，在交叉、突变遗传的过程中，优秀的结构会随着基因遗传到子代，进而使得搜索向较优秀的个体进行；
    b. 同时，遗传算法具有较好的鲁棒性，以及较强搜索能力，而组合优化大部分搜索空间相对较小，使得得遗传算法能够搜索大部分解，得到最优结果。
2. 遗传算法不太能适用于神经网络的原因包括：
    a. 神经网络中得权重信息不能较好地或者可解释性地嵌入到编码中，这使得这种杂交遗传不一定利用了或者充分利用了问题得启发式信息，使得搜索效果不好。
    b. 在大规模神经网络中，权重参数较多，这使得编码较长，问题的解空间非常大，而遗传算法是启发式算法，虽然具有很强的搜索能力，但是随机性也存在，这样使得在搜索过程中，不仅不一定随着最优解的方向进行，而且基因交叉和突变带来的好的影响相对就削弱了。

c. 遗传算法是大规模的随机搜索算法，在实际过程中，由于神经网络参数量大，这种随机搜索的代价开销较大，同时由于其收敛性问题，可能一直震荡无法收敛，相比较于梯度下降、SGD、Adam等有数学基础的有贪心性质的搜索方法，其劣势是较大的，前者带来的开销是更加能接受的。

d. 在现代大数据大网络的情况下，可以发现，神经网络的参数初始值对最终的效果影响不大，最后基本均能收敛到同一个值，因此使用GA进行参数初始化是没有太大必要的。

e. 因此从实际开销来看，可能GA对NN初始化、直接更新参数均没有特别好的效果；使用梯度信息更加利于网络训练

3. 在GA算法中，参数设置非常重要：

   要平衡探索与利用：

   - 种群规模代表了一定的搜索范围，其设定要适当；迭代次数也会影响GA的全局搜索能力，因此可以较大，但是要注意开销；而突变算子中突变的概率也要设置适当，不能太小，因为搜索过程相比较实际进化是很短的，突变概率要适当大，以增强后期的局部搜索能力，但是不能过大而影响全局搜索。
   - 需要设置较好的参数使得前期注重全局搜索能力，向最优解靠近；同时在后期注重局部搜索，找到最优解；
   - 因此可以进行如下改进：设置突变概率和交叉概率：前者逐渐随迭代次数变大，后者随迭代次数减小，以拟合探索与利用的规律，在后期，可以更加贪心，在当前最优解附近继续搜索更好的解

4. GA的算法设计特别是编码中，一定要较好地嵌入问题信息，充分利用启发式信息；在遗传算子的设计中，也要充分考虑下如何保留父代的优秀信息，使得种群的优秀基因和信息可以遗传，逐代变好。

九.遇到的问题于解决方法

1. 算力问题：在神经网络实验中，由于算力问题选择了参数初始化的实验；选用paddle库搭建神经网络，这样可以在百度平台上进行GPU使用。

2. 代码问题：在神经网络实验中，使用paddle库并不熟练，在如何用指定的参数初始化线性层上遇到了问题，最后通过阅读paddle的API文档，寻找相应的函数以及说明，将问题解决了。

```
se:#如果输入了初始化参数，使用初始化参数
  weight_in2hid=paddle.framework.ParamAttr(initializer=nn.initializer.Assign(weight[0:feature*hidden].reshape([feature,hidden])
  bias_in2hid=paddle.framework.ParamAttr(initializer=nn.initializer.Assign([weight[feature*hidden:feature*hidden+hidden]]))
  self.input_layer=nn.Linear(in_features=feature,out_features=hidden,weight_attr=weight_in2hid,bias_attr=bias_in2hid)
  weight_hid2out=paddle.framework.ParamAttr(initializer=nn.initializer.Assign(weight[feature*hidden+hidden:feature*hidden+hidde
  bias_hid2out=paddle.framework.ParamAttr(initializer=nn.initializer.Assign([weight[feature*hidden+hidden+hidden*class_dim:]]))
  self.hidden_layer=nn.Linear(in_features=hidden,out_features=class_dim,weight_attr=weight_hid2out,bias_attr=bias_hid2out)
```

   在python语法中许多都是通过指针指向同一个对象的赋值方法，因此在赋值时，需要注意可能需要循环逐个赋值，或者使用copy库进行赋复制拷贝赋值。

3. 在度约束的MST中，遇到的问题是编码以及交叉如何设计，通过查阅文献，解决了此问题。
4. 同时，在如何解码生成树中遇到了问题，在解码过程中，自己写的算法会会生成森林，不符合树的条件；通过观察发现，在初始化后的过程中，只要初始中群能够解码生成树，那么在交叉和突变中均不影响其 解码，所以在初始化种群是增加了test函数，观察生成的个体是否可以正确解码，来排除非法个体，使得问题得到解决。

```
def test_function(self,entity):#观察子代是否满足条件
    eages=self.decoding_function(entity[0],entity[1])
    if len(eages)==self.n-1:
        return True
    else:
        return False
```

5. 最后自己的一个设想还未进行实验，可以空余时间进行相关实验，使用改进的算法进行度约束的MST问题，验证猜想的正确性。

十.收获与感想

- 首先，通过阅读相关文献，基本了解了遗传算法的应用领域；同时在神经网络的优化余度约束的最小生成树问题的实践过程中，通过阅读文献了解了设计细节，结合课堂知识，总结出了遗传算法的重要设计部分，通过代码实践，对每一步的原理进一步有了理解和掌握。最后通过实验结果，纵向和横向比较：
    - 得出了遗传算法不适用于神经网网络的主要原因可能是其随机性，不收敛的问题，以及在数据较大时，参数初始化对最终模型影响效果不大，同时遗传算法是分布式的随机搜索算法，代价较大，在实际过程中，这也是影响因素之一。
    - 遗传算法更适用于组合优化的问题，由于其离散的、解释性较强的编码方式，其交叉变异遗传更具有可解释性，如：父代的好的基因在遗传给子代后，实际上是遗传了较好的树的结构，这实际就是在通过全局搜索逐渐向最优解移动，然后通过变异的方式，加强局部搜索，最终得到最优解；这种良好的“探索与利用”的平衡让遗传算法在这类优化问题中有较好的表现。
- 通过python代码完成了实验任务，实现了遗传算法、神经网络的搭建、深度优先遍历树等，进一步加深了我对算法原理的理解，同时夯实了我的代码实践能力，让我对相关函数、库的学习了解更多，为以后的算法代码实践有一定帮助。
- 最后，通过本次实验，自行查找资料文献、阅读相关项目代码，提高了自己的问题求解能力和独立处理问题的能力。

十一.参考文献

[1].董军, 关凤岩, 吕宗宝. 基于遗传算法度约束的最小生成树问题的研究[J]. 淮北煤炭师范学院学报：自然科学版,2005, 26(1):4.

[2].余皓然.最优化方法课件-实数编码的仿二进制编码交叉.