华北电力大学

**实 验 报 告**

|

|

**实验名称**  **PROLOG图搜索问题求解及计算智能算法设计**

**课程名称 人工智能**

专业班级：信安1802 学生姓名：王楚璇

学 号：2018090000118 成 绩：

指导教师：刘丽 实验日期：2020.12

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **一、实验目的及要求**  **实验内容如下：**   1. 基于PROLOG的逻辑程序设计。   熟悉PROLOG程序设计环境，掌握基于PROLOG的程序设计方法，选做以下题目之一：路线问题、家谱问题、险洞探险问题、八数码问题、八皇后问题、农夫过河问题或传教士与野人问题。   1. 计算智能相关算法设计。   熟悉遗传算法和人工神经网络的基本原理，选做以下题目之一：基于遗传算法的函数极值求取、基于感知器的数据线性分类。  **基本要求：**   1. 掌握基于PROLOG进行程序设计的方法和步骤，在实验报告中回答指定的专门问题。 2. 基于PROLOG的逻辑程序设计。如选做路线问题、家谱问题、险洞探险问题之一，要对待求解的问题增加复杂度，如扩充地图，扩充关系，增加求解要求等，要体现对问题的深入分析，体现新颖性；如选做八数码问题、八皇后问题、农夫过河问题和传教士与野人问题之一，要通过对程序的调试掌握基于PROLOG进行图搜索的基本方法。 3. 计算智能相关算法设计。编程语言自定，可选择MATLAB、C++、Java等；基于遗传算法的函数极值求取要求能自行制定待求解的函数，运行结果能体现极值，迭代次数，种群规模等；基于感知器的数据线性分类可实现逻辑与、逻辑或、或者实现对下述问题的简单分类。   训练数据   |  |  |  |  | | --- | --- | --- | --- | | 类型 | 鲜红值 | 直径 | 质量 | | 车厘子 | 0.81 | 1.02 | 8.85 | | 车厘子 | 0.82 | 0.98 | 8.67 | | 车厘子 | 0.78 | 0.99 | 8.75 | | 车厘子 | 0.79 | 1.01 | 8.80 | | 樱桃 | 0.56 | 0.85 | 7.32 | | 樱桃 | 0.58 | 0.86 | 7.33 | | 樱桃 | 0.59 | 0.83 | 7.29 | | 樱桃 | 0.57 | 0.84 | 7.31 |   测试数据   |  |  |  |  | | --- | --- | --- | --- | | 类型 | 鲜红值 | 直径 | 质量 | |  | 0.60 | 0.88 | 7.45 | |  | 0.76 | 1.00 | 8.78 |   **二、所用仪器、设备**  PC机、Trinc prolog编译软件和其他所需软件等。  **三、实验原理**  基于PROLOG的逻辑程序设计实验利用了prolog程序强大的模式匹配与递归功能。家谱问题主要体现了prolog程序模式匹配功能，首先将事实和相应的血缘关系规则列出，当向prolog程序询问某两个人的关系时，prolog会根据问题自动进行关系匹配、回溯，将匹配到的结果输出。  传教士与野人问题我觉得主要体现了prolog程序的递归功能，最能体现的便是findroad函数。首先输入初始状态和最终状态，findroad函数就会自动进行递归，每次执行时都会判断是否符合题目预先给的条件，如果不符合，进行回溯，如果符合，进行路径输出，直至找到最终的目标。  基于遗传算法的函数极值求取实验，语言方面采用了python，因为python有很方便的作图功能。遗传算法模拟自然选择和自然遗传过程中发生的繁殖、交叉和基因突变现象，在每次迭代中都保留一组候选解，并按某种指标从解群中选取较优的个体，利用遗传算子(选择、交叉和变异)对这些个体进行组合，产生新一代的候选解群，重复此过程，直到满足某种收敛指标为止。其遗传进化操作过程简单，容易理解。  基于感知器的数据线性分类实验，语言方面采用了python。该程序原理是基于感知器的线性分类，首先对一组数据进行训练，通过激励函数的变换，得到预期结果，根据预期结果与实验结果的误差，不断修改权重值，直至最后权重收敛，结束计算，得到满意的计算结果。  **四、实验方法与步骤**  **验证性实验：**  通过该实验掌握了prolog的简单程序结构，掌握分析问题、询问解释技巧；进行事  实库、规则库的编写，并在此基础上进行简单的询问。  **家谱问题：**  在原来的基础上，我以我家家谱为模型，原来的关系上添加了gongong(公公，丈夫的父亲),popo(婆婆，丈夫的母亲),yuefu(岳父，妻子的父亲),yuemu(岳母，妻子的母亲),bofu(伯父，父亲的哥哥),bomu(伯母，父亲的哥哥的妻子),shushu(叔叔，父亲的弟弟),shenshen(婶婶，父亲的弟弟的妻子),dajiu(大舅，母亲的哥哥),dajiuma(大舅妈，母亲的哥哥的妻子),xiaojiu(小舅，母亲的弟弟),xiaojiuma(小舅妈，母亲的弟弟的妻子),daguma(大姑妈，父亲的姐姐),daguzhang(大姑丈，父亲的姐姐的丈夫),xiaoguma(小姑妈，父亲的妹妹),xiaoguzhang(小姑丈，父亲的妹妹的丈夫),dayi(大姨，母亲的姐姐),dayifu(大姨父，母亲的姐姐的丈夫),xiaoyi(小姨，母亲的妹妹),xiaoyifu(小姨夫，母亲的妹妹的丈夫),cousin(堂/表兄妹),dajiuzi(大舅子，妻子的哥哥),xiaojiuzi(小舅子，妻子的弟弟),dayizi(大姨子，妻子的姐姐),xiaoyizi(小姨子，妻子的妹妹),dabozi(大伯子，丈夫的哥哥),xiaoshuzi(小叔子，丈夫的弟弟),daguzi(大姑子，丈夫的姐姐),xiaoguzi(小姑子，丈夫的妹妹),saozi(嫂子，哥哥的妻子),dimei(弟妹，弟弟的妻子),jiefu(姐夫，姐姐的丈夫),meifu(妹夫，妹妹的丈夫),zhizi(侄子，兄弟的儿子),zhinv(侄女，兄弟的女儿，丈夫的兄弟的女儿),waisheng(外甥，姐妹的儿子),waishengnv(外甥女，姐妹的女儿，丈夫的姐妹的女儿),neizhi(内侄，妻子的兄弟的儿子),neizhinv(内侄女，妻子的兄弟的女儿),neisheng(内甥，妻子的姐妹的儿子),yishengnv(姨甥女，妻子的姐妹的女儿),shuzhi(叔侄，丈夫的兄弟的儿子),gusheng(姑甥，丈夫的姐妹的儿子)。名称是根据网络上的亲戚计算器算得，只列出我的家谱中用到的部分。根据此程序可以较全面的反映两个人之间的关系，得出“称呼”。  图中a1,a2后代为b2,b4,b5.b7（年龄按次序从大到小），其中蓝色代表男性，粉色代表女性。    **传教士与野人：**  本实验主要的思想就是无论在任何时候，两岸的野人数量都不能大于传教士的数量，在这个前提下，通过给出的移动的实例move(X,Y)，采用递归函数findroad，并且在函数里面调用connect、update以及member函数，不断试探从初始状态到最终状态的路径，最后得到完整的过河方法。  **遗传算法：**  本实验遗传算法的构成要素：染色体编码方法，个体适应度评价，遗传算子，基本遗传算法的运行参数。  首先对于预解决问题的优化变量进行某种形式的编码，再确定问题的适应度函数，本实验是自定函数（设置成二元函数x1,x2），进行极值求解，将群体中的个体代入目标函数，计算个体的适应度值，由计算出的适应度值评价染色体的优劣，若满足精度或达到最大迭代次数则输出解，否则迭代次数加一，染色体进行遗传操作，以一定的方式选择一定数量的较优个体进行交叉操作，对于交叉完的部分染色体进行变异操作，而后再重复前面的操作。最后得出问题的解。  **感知器：**  确定初始化参数；搭建感知器模型：对一组数据进行训练，通过激励函数的变换，得到预期结果，根据预期结果与实验结果的误差；利用反向算法，完成权重系数的调整：不断修改权重值，直至最后权重收敛，结束计算，得到满意的计算结果。本实验的感知器能实现对车厘子樱桃的简单分类。  **五、求解的问题与程序**  **家谱问题：**  求解：  ?- dimei(X,Y).  输出结果：  SOLUTION:  X=b6  Y=b2  SOLUTION:  X=b6  Y=b4  SOLUTION:  X=b8  Y=b2  SOLUTION:  X=b8  Y=b4  SOLUTION:  X=b8  Y=b5  no  求解：  ?-saozi(X,Y).  输出结果：  SOLUTION:  X=b6  Y=b7  no  主要代码如下（代码较多，给出部分）：  grandpa(X,Y):-father(X,Z),mother(Z,Y).  grandma(X,Y):-mother(X,Z),mother(Z,Y).  grandfather(X,Y):-father(X,Z),father(Z,Y).  grandmother(X,Y):-mother(X,Z),father(Z,Y).  gongong(X,Y):-father(X,Z),husband(Z,Y).  popo(X,Y):-mother(X,Z),husband(Z,Y).  yuefu(X,Y):-father(X,Z),wife(Z,Y).  yuemu(X,Y):-mother(X,Z),wife(Z,Y).  bofu(X,Y):-elder\_brother(X,Z),father(Z,Y).  bomu(X,Y):-wife(X,Z),bofu(Z,Y).  shushu(X,Y):-younger\_brother(X,Z),father(Z,Y).  shenshen(X,Y):-wife(X,Z),shushu(Z,Y).  dajiu(X,Y):-elder\_brother(X,Z),mother(Z,Y).  dajiuma(X,Y):-wife(X,Z),dajiu(Z,Y).  xiaojiu(X,Y):-younger\_brother(X,Z),mother(Z,Y).  xiaojiuma(X,Y):-wife(X,Z),xiaojiu(Z,Y).  dajiuzi(X,Y):-elder\_brother(X,Z),wife(Z,Y).  xiaojiuzi(X,Y):-younger\_brother(X,Z),wife(Z,Y).  dayizi(X,Y):-elder\_sister(X,Z),wife(Z,Y).  xiaoyizi(X,Y):-younger\_sister(X,Z),wife(Z,Y).  dabozi(X,Y):-elder\_brother(X,Z),husband(Z,Y).  xiaoshuzi(X,Y):-younger\_brother(X,Z),husband(Z,Y).  daguzi(X,Y):-elder\_sister(X,Z),husband(Z,Y).  xiaoguzi(X,Y):-younger\_sister(X,Z),husband(Z,Y).  saozi(X,Y):-wife(X,Z),elder\_brother(Z,Y).  dimei(X,Y):-wife(X,Z),younger\_brother(Z,Y).  jiefu(X,Y):-husband(X,Z),elder\_sister(Z,Y).  meifu(X,Y):-husband(X,Z),younger\_sister(Z,Y).  zhizi(X,Y):-father(Z,X),elder\_brother(Z,Y),man(X).  zhizi(X,Y):-father(Z,X),younger\_brother(Z,Y),man(X).  zhinv(X,Y):-father(Z,X),elder\_brother(Z,Y),woman(X).  zhinv(X,Y):-father(Z,X),younger\_brother(Z,Y),woman(X).  waisheng(X,Y):-mother(Z,X),elder\_sister(Z,Y),man(X).  waisheng(X,Y):-mother(Z,X),younger\_sister(Z,Y),man(X).  waishengnv(X,Y):-mother(Z,X),elder\_sister(Z,Y),woman(X).  waishengnv(X,Y):-mother(Z,X),younger\_sister(Z,Y),woman(X).  **传教士野人问题：**  move(1,0).%%移动一个传教士，0个野人  move(0,1).%%移动0个传教士，1个野人  move(0,2).%%移动0个传教士，2个野人  move(2,0).%%移动2个传教士，0个野人  move(1,1).%%移动1个传教士，1个野人  **%%判断此时的状态是否合法**  legal((X,Y,\_)):-ok(X), ok(Y).  ok((X,Y)):-X=0,Y>=0,!.**%%传教士数量为零，野人数量大于零，合法**  ok((X,Y)):-Y=0,X>=0,!.**%%传教士数量大于零，野人数量等于零，合法**  ok((X,Y)):-X>=Y,X>=0,Y>=0.**%%传教士数量大于野人的数量，合法**  update((X,Y,0),Move,S1):- (A,B)=X,(C,D)=Y,(E,F)=Move,C1 is C+E, D1 is D+F, A1 is A-E,B1 is B-F,S1=((A1,B1),(C1,D1),1).  **%%进行状态更新，将船在右岸时的状态（(A,B),(C,D),0）变为船在左岸时的状态S1=((A1,B1),(C1,D1),1)，在这个过程中，从右岸向左岸运送E个传教士，F个野人**  update((X,Y,1),Move,S1):- (A,B)=X,(C,D)=Y,(E,F)=Move,C1 is C-E, D1 is D-F,A1 is A+E,B1 is B+F,S1=((A1,B1),(C1,D1),0).  **%%进行状态更新，将船在左岸时的状态（(A,B),(C,D),1）变为船在右岸时的状态S1=((A1,B1),(C1,D1),0)，在这个过程中，从左岸向右岸运送E个传教士，F个野人**  connect(Statu,S1):-move(X,Y),update(Statu,(X,Y),S1),legal(S1).  **%%判断是否能从Statu状态，移动X个传教士，移动Y个野人，转换到S1状态**  member(X,[X|Tail]).  member(X,[Head|Tail]):-member(X,Tail).**%%判断X是不是表中的元素**  findroad(X,X,L,L):- write(L).**%%递归的边界条件，如果两个状态相同，那就直接输出路径L**  findroad(X,Y,L,L1):-connect(X,Z),not(member(Z,L)), findroad(Z,Y,[Z|L],L1).**%%如果两个状态不同，通过connect函数，找到中间路径Z，并且Z不是L中的点，那么进行递归，把Z代入findroad进行递归**  **%L为存储的路由表**  **遗传算法：**  基于遗传算法的函数极值求取自行制定待求解的函数。  求解函数：    主要代码如下：  def selectOp():  global population  cfitness = []  sum = 0  newPopulation = []  for i in range(popSize):  sum += population[i].fitness  for i in range(popSize):  cfitness.append(population[i].fitness / sum)  for i in range(1, popSize):  cfitness[i] = cfitness[i - 1] + cfitness[i]  for i in range(popSize):  p = random.random()  index = 0  while p > cfitness[index]:  index += 1  newPopulation.append(copy.deepcopy(population[index]))  for i in range(popSize):  population[i] = newPopulation[i]  def corssoverOp(): # 交叉  global population  ###########################################  index = []  for i in range(popSize):  index.append(i)  for i in range(popSize):  point = random.randint(0, popSize - 1)  temp = index[i]  index[i] = index[point]  index[point] = temp  for i in range(0, popSize - 1, 2):  p = random.random()  if p < Pc:  point = random.randint(0, chromLength - 1)  for j in range(point, chromLength):  temp = population[index[i]].chrom[j]  population[index[i]].chrom[j] = population[index[i + 1]].chrom[j]  population[index[i + 1]].chrom[j] = temp  def mutationOp(): # 变异  global population  for i in range(popSize):  for j in range(chromLength):  p = random.random()  if p < Pm:  population[i].chrom[j] = 0 if population[i].chrom[j] == 1 else 1  **感知器：**  实现对下述问题的简单分类。  训练数据   |  |  |  |  | | --- | --- | --- | --- | | 类型 | 鲜红值 | 直径 | 质量 | | 车厘子 | 0.81 | 1.02 | 8.85 | | 车厘子 | 0.82 | 0.98 | 8.67 | | 车厘子 | 0.78 | 0.99 | 8.75 | | 车厘子 | 0.79 | 1.01 | 8.80 | | 樱桃 | 0.56 | 0.85 | 7.32 | | 樱桃 | 0.58 | 0.86 | 7.33 | | 樱桃 | 0.59 | 0.83 | 7.29 | | 樱桃 | 0.57 | 0.84 | 7.31 |   测试数据   |  |  |  |  | | --- | --- | --- | --- | | 类型 | 鲜红值 | 直径 | 质量 | |  | 0.60 | 0.88 | 7.45 | |  | 0.76 | 1.00 | 8.78 |   主要代码如下：  def activation():  global b  global e  for i in range(len(a)):  b[i]=a[i][0]\*w[0]+a[i][1]\*w[1]+a[i][2]\*w[2]  b[i]=b[i]-sita  if b[i]>=0:  b[i]=1  else:  b[i]=0  e[i]=a[i][3]-b[i]  def train():  global w  for i in range(len(a)):  for j in range(3):  deltaWi=alpha\*a[i][j]\*e[i]  w[j]=w[j]+deltaWi  def judge():  for i in range(len(e)):  if e[i]!=0:  return False  return True  if \_\_name\_\_=='\_\_main\_\_':  flag=0  round=0  init()  while flag==0:  activation()  train()  round+=1  if round>=10000:  flag=1  if judge():  break  if flag==1:  print('他不收敛')  else:  for i in range(len(aa)):  bb[i] = aa[i][0] \* w[0] + aa[i][1] \* w[1] + aa[i][2] \* w[2]  bb[i]=bb[i]-sita  if bb[i]>=0:  bb[i]=1  else:  bb[i]=0  if bb[i]==1:  print('樱桃')  else:  print('车厘子')  **六、讨论与结论**  家谱问题中，由于我用自己家庭作为模型，并尽可能还原了亲戚之间的称呼，所以复杂度会有一点高，但是难度并不是很大，只需注意各个人之间的关系即可。传教士野人问题中，问题本身并不是很难，但是用prolog语言将其形式化表达出来还是有一定的难度，我参考了老师给的样例，用自己的想法将其重新表达出来。遗传算法问题整体的工作量较多，我使用了python语言，一方面是它简单清晰，另一方面是有作图功能。在编写过程中，我发现每一代作一次图较不直观且费时，我就改成每十代输出一次图像。一开始引用导致每一代整体数量都变少，后来采用deepcopy以后深复制才解决这个方法。感知器实验较遗传算法简单，但是中间由于一些参数设置的不合理导致结果输出有异议，后来修改参数基本解决了问题。  遗传算法的制定待求解的函数设置为二元函数，中间每隔10代输出一次，更直观的表现出来了遗传算法的特点，其中图像的x轴和y轴分别代表两个元x1,x2，其变化反映了染色体分布变化。感知器算法简单易懂，设置了最大迭代次数，使算法更加完整。  **13.Prologue发展历史：**  Prolog这个名字是源自用逻辑编程的英文词缩写：Pro-gramming in Log-ic。 在20世纪60年代末的时候Aix-Marseille大学的Alain Colmerauer和 Philippe Roussel在Robert Kowalski的工作基础上（霍恩子句的过程化描述），开始了一门新的编程语言的创立。最终在1972年，正式宣布创立Prolog编程语言。该编程语言建立在一阶谓词逻辑的基础之上，有强烈的数学风格，和我们目前比较熟悉的面向对象的思路相比，有非常明显的差异。自1972年Prolog发布以后，分支出多种Prolog的方言。最早的Prolog解释器由Roussel建造，而第一个Prolog编译器则是 David Warren编写的。而在此之后，Prolog被全世界大范围的使用，日本政府曾经为了建造智能计算机而用Prolog来开发ICOT第五代计算机系统。在早期的机器智能研究领域，Prolog曾经是主要的开发工具。80年代Borland开发的Turbo Prolog，进一步普及了Prolog的使用。1995年确定了ISOProlog标准。  **七、所附实验输出的结果或数据**  **家谱问题：**    **遗传算法：**        **感知器：** |
|  |