**人工智能作业 第三次**

**搜索问题形式化**

题目：将一个实际问题建模为搜索问题。 评分标准和要求： （1）实际问题（来自老师的科研）、新颖问题（生活中遇到的复杂问题）、自己感兴趣的问题，越少人知道或了解的问题，评分越高。（2）形式化描述问题（搜索问题5要素讲清楚）（3）状态空间的大小分析清楚（4）基准搜索算法，及简单的时空性能分析；（5）设计两个启发函数，并比较二者优劣。

1. **实际问题**

在进行超算任务运行时间预测的时候有一个重要的步骤就是要确定能够定义运行任务相似的特征。以往的做法都是直接人为的预先定义。研究者先预先选择几个特征，然后在这些特征上，比较两个任务的相似度，但是这种方法有很大的局限性，如果我们预先选择的特征参数不是最优的话，将会对结果有很大的影响。因此，我们可以将寻找相似度比较最佳参数的过程看做是一个搜索过程，将最优的参数特征列表作为搜索的结果。

1. **形式描述**

状态：一个由若干个特征组成的特征列表

初始状态：不包含任何特征的空特征列表

后继函数：加入或不加入一个新的特征

目标测试：找到具有最小的均方误差的特征列表

路径耗散：加入或不加入一个特征，代价为1

1. **状态空间的大小分析**

假设一个超算任务可以用n维的特征向量来描述，那么它的状态空间就是2^n

1. **基准搜索算法和时空性能分析**

基准搜索算法：

输入: 状态图(G), 初始状态S0

输出：具有最小的均方误差的特征列表

过程：

While 存在还未遇到的状态：

计算在当前状态下，所有任务相似度的均方误差

找到当前状态的所有后继状态

对于后继状态中还没遇到过的状态：

将此状态作为当前状态，进行递归

对于后继状态中已经遇到过的状态：

直接忽略

从所有的结果中找到具有最小的均方误差的特征集合，就是我们想要的解

时空性能分析：它的时空性能复杂性为n\*(n-1)\*(n-2)\*......\*1 = n!

1. **两个启发函数设计及优劣比较**

对于一个有n维特征的超算任务

1. 采用贪心算法：
2. 令解集 T={()}
3. for i = 1 to n:
   1. 令 Tc 包含 n/i个不同局部解，每个局部解有i个特征
   2. For Tc中的每个局部解tc：
      * + 1. 创建一个临时解集Xc= T U {tc}
          2. 算出Xc在所有超算任务中的均方误差
   3. 选择具有最小均方误差值的Xc,并把相关的tc加入到T
4. 得到解T={()(h1,1)(h2,1，h2,2).....(hi,1，hi,2，......，hi,i)}
5. 然后就可以根据需要从解集中获取想要的最优解
6. 采用遗传算法
7. 初始个体是一些包含1到10局部解的解集
8. 按照以下的编码规则对解集尽心编码：
   1. 是否采用均值或者线性回归
   2. 是否使用绝对火相对运行时间
   3. 是否加入了每个和任务相关的二值特征
   4. 是否使用了节点信息
9. 适应度函数是：

Fmin + (Emax-E)/(Emax-Emin).(Fmax-Fmin)

其中E是个体错误，Emax,Emin是在遗传中个体的最大和最小错误，Fmin和Fmax是想要的最小和最大适应度。设定Fmax = 4\*Fmin

1. 采用启发式抽样进行个体选择，每个个体都有F/Favg的机会成为parent，剩下的parents则采用伯努利分布进行选择
2. 交叉运算的定义：随机的选择一对个体进行交叉运算，然后用他们的后代代替他们。在交叉运算的过程中也使用了精英原则，对于最好的个体，我们不对他进行交配，而是让它直接进入下一代选择
3. 当迭代遗传的次数达到我们预先定义的值，就停止。找到其中最好的个体就是我们想要的解
4. 两个启发式函数的比较
   1. 前者的算法比较简单，运算的时间复杂度比较小。后者的算法复杂，不仅需要预先定义适应度函数，交叉运算函数等，还需要预先定义合适的迭代次数。
   2. 前者的算法可以搜索任意的特征解集。后者的算法则不行。它的解集空间很大程度上取决于定义的初始个体。如果初始个体选的不好，那么无论迭代多少次，效果都不会好。
   3. 前者算法很容易就陷入了局部最优。后者算法一般在适度函数选择不当的时候才有可能陷入局部最优。