## 算法设计与分析第5章作业

计算机72班 徐子越 2174410455

5-4>设计一个解最大团问题的迭代回溯法

解：

1、解决思路：

最大团指的是一个图的最大完全图，根据回溯法以及组合数学的知识我们可以用排列树来遍历解空间，用相应的剪枝方法来优化复杂度。

解决细节是将最大团问题转为二叉树，按照组合数学中的组合排列数来遍历。

2、C++代码如下：

#include <iostream>  
#include <vector>  
using namespace std;  
int main(){  
 int n = 5;  
 int count = 0;  
 int best\_res\_num = 0;  
 vector<int> present\_answer = {0,0,0,0,0};  
 vector<int> end\_answer = {1,1,1,1,1};  
 vector<int> best\_answer;  
 vector<vector<int>> acc\_arr(n);  
  
 for(int i = 0;i < n;i++)  
 acc\_arr[i].resize(n);  
 acc\_arr[0]={1,1,0,1,1};  
 acc\_arr[1]={1,1,1,0,1};  
 acc\_arr[2]={0,1,1,0,1};  
 acc\_arr[3]={1,0,0,1,1};  
 acc\_arr[4]={1,1,1,1,1};  
  
 while(present\_answer != end\_answer){  
 int p = 0;  
 int sign = 0;  
 for(int i = 0;i<n;i++)//找出当前数组的最低位置的0的位置p  
 if(present\_answer[i]==0){  
 p = i;  
 //cout<<p;  
 break;  
 }  
 for(int i = 0;i<n;i++){//剪枝，如果应该被剪则sign标记为1,否则为0  
 if(present\_answer[i]==1)  
 for(int j = 0;j<n;j++)  
 if(present\_answer[j]==1){  
 if(acc\_arr[i][j]==0){  
 sign = 1;  
 break;  
 }  
 }  
 }  
 if(sign == 0){//如果sign为0,则检测再加上，并且与当前最佳答案比较，更佳者覆盖  
 count = 0;  
 for(int i = 0;i<n;i++) {  
 if (present\_answer[i] == 1)  
 for (int j = 0; j < n; j++)  
 if (present\_answer[j] == 1) {  
 if (acc\_arr[i][j] == 0) {  
 count = 1;  
 break;  
 }  
 }  
 }  
 if(count==0){  
 int cmp = 0;  
 for(int i=0;i<n;i++){  
 if(present\_answer[i]==1)  
 cmp++;  
 }  
 if(best\_res\_num<cmp){  
 best\_answer = present\_answer;  
 best\_res\_num = cmp;  
 }//覆盖最佳答案  
 }  
 for(int i = 0;i<n;i++)  
 if(present\_answer[i]==0){  
 present\_answer[i] = 1;  
 break;  
 }  
 else if((i==n-1)&&(present\_answer[i]==1))  
 present\_answer = end\_answer;  
 else if(present\_answer[i]==1){  
 present\_answer[i] = 0;  
 }  
 }  
 else{//剪枝跳过剩下的部分，将数组p后续置为0,前面的进一位  
 for(int i = p;i>=0;i--)  
 present\_answer[i] = 0;  
 if(p!=n-1)  
 for(int i = p+1;i<n;i++){  
 if(present\_answer[i]==0){//p之后全部置为0  
 present\_answer[i] = 1;  
 break;  
 }  
 else if((i==n-1)&&(present\_answer[i]==1))  
 present\_answer = end\_answer;  
 else if(present\_answer[i]==1)  
 present\_answer[i] = 0;  
 }  
 else{  
 present\_answer[p]=1;  
 for(int i = 0;i<n-1;i++)  
 present\_answer[i] = 0;  
 }  
 }  
 }  
 for(int i=0;i<5;i++)  
 cout<<best\_answer[i]<<" ";  
 return 0;  
}

1. 运行结果：



5-6>解：

（1）显而易见，前缀为1～i的回路费用为式子前部分，后半部分为i～n所走过的路的最小值，为每个地点出发的最小边之和。

（2）在旅行售货员问题中采用（1）中的剪枝函数即如果已经记录的最佳费用比当前的路线的最小费用还要小，那么这条路就可以剪掉了。

算法：

设n0为初始点

选择n0的所有可达城市n1，记录费用

再选择n1的子城市，记录费用

直到到达终点，记录下总费用

回到n0选择下一个子城市，将上述在再做一边，记录费用

如果在途中最小剩余费用已经超过最佳记录费用，那么剪枝，否则继续

直到最后记录最后费用，与最佳费用比较更新最佳费用。

其中，城市的便利使用树的中序遍历

与书上原算法对比新增了一个剪枝检测，复杂度降低。