①算法分析题5-4 试设计一个解最大团问题的迭代回溯法。

首先设最大团为一个空团，往其中加入一个顶点，然后依次考虑每个顶点，查看该顶点加入团之后仍然构成一个团，如果可以，考虑将该顶点加入团或者舍弃两种情况，如果不行，直接舍弃，然后递归判断下一顶点。对于无连接或者直接舍弃两种情况，在递归前，可采用剪枝策略来避免无效搜索。

为了判断当前顶点加入团之后是否仍是一个团，只需要考虑该顶点和团中顶点是否都有连接。

程序中采用了一个比较简单的剪枝策略，即如果剩余未考虑的顶点数加上团中顶点数不大于当前解的顶点数，可停止继续深度搜索，否则继续深度递归。

代码如下：

#include <iostream>

using namespace std;

class Clique

{

friend void MaxClique( int \*\*, int \*, int );

private :

void Backtrack( int i);

int \*\*a; //图的邻接矩阵

int n; //图 的顶 点数

int \*x; //当前解

int \*bestx; //当前最优解

int cn; //当前顶点数

int bestn; //当前 最 大顶点数

};

void Clique::Backtrack( int i)

{ //计算最大团

if (i>n) //到达叶子节点

{

for ( int j=1;j<=n;j++)

bestx[j]=x[j];

bestn=cn;

cout<< "最大团：（" ;

for ( int i=1;i<n;i++)

cout<<bestx[i]<< "," ;

cout<<bestx[n]<< ")" <<endl;

return ;

}

//检 查 当前顶点是否与当前团连接

int ok=1;

for ( int j=1;j<i;j++)

if (x[j]&&a[i][j]==0) //i与j不连接,即j在团中,但是i,j不连接

{

ok=0;

break ;

}

if (ok) //进入左子树

{

x[i]=1;

cn++;

Backtrack(i+1); //回溯到下一层节点

x[i]=0;

cn--;

}

//通过上界函数判断是否减去右子树，上界函数用于确认还有足够多的可选择顶点使得算法有可能在右子树中找到更大的团

if (cn+n-i>=bestn)

{ //修改一下上界函数的条件，可以得到

x[i]=0; //相同点数时的解

Backtrack(i+1);

}

}

void MaxClique( int \*\*a, int \*v, int n)

{ //初始化 Y

Clique Y;

Y.x= new int [n+1];

Y.a=a;

Y.n=n;

Y.cn=0;

Y.bestn=0;

Y.bestx=v;

Y.Backtrack(1);

delete [] Y.x;

cout<< "最大团的顶点数：" <<Y.bestn<<endl;

}

int main()

{

int n;

cout<< "please input number of node:" ;

cin>>n;

//int a[n+1][n+1]; //由于定义的是int \*\*a，且采用的是二维数组传参，因此

int \*\*a= new int \*[n+1]; //两种解决方法，一是给定第二维的大小，二是通过

for ( int k=0;k<=n;k++) //动态分配内存，这里采用了动态内存分配解决问题

a[k]= new int [n+1];

for ( int i=0;i<n+1;i++)

for ( int j=0;j<n+1;j++)

a[i][j]=0;

int edge;

cout<< "please input number of edge:" ;

cin>>edge;

cout<< "please input edge:" <<endl;

int v,w;

for ( int x=0;x<edge;x++)

{

cin>>v>>w;

a[v][w]=1;

a[w][v]=1;

}

int \*p= new int [n+1];

MaxClique(a,p,n);

system( "pause" );

return 0;

}

②算法实现题5-4 运动员最佳配对问题

问题描述：羽毛球队有男女运动员各n人。给定2个n×n矩阵P和Q。P[i][j]门是男运动员i和女运动员j配对组成混合双打的男运动员竞赛优势；Q[i][j]是女运动员i和男运动员j配合的女运动员竞赛优势。由于技术配合和心理状态等各种因素影响，P[i][j]不一定等于Q[i][j]。男运动员i和女运动员j配对组成混合双打的男女双方竞赛优势为P[i][j] \* Q[i][j]。设计一个算法，计算男女运动员最佳配对法，使各组男女双方竞赛优势的总和达到最大。

编程任务：设计一个算法，对于给定的男女运动员竞赛优势，计算男女运动员最佳配对法，使各组男女双方竞赛优势的总和达到最大。

解：代码如下：

#include<bits/stdc++.h>

using namespace std;

int n;

int p[100][100];

int q[100][100];

int x[100];

int best[100];

int answer=0;

void swap(int &a,int &b){

int temp;

temp=a;

a=b;

b=temp;

}

void update(){

int sum=0;

for(int i=1;i<=n;i++){

sum+=p[i][x[i]]\*q[x[i]][i];

}

if(sum>answer){

answer=sum;

for(int i=1;i<=n;i++){

best[i]=x[i];

}

}

}

void backtrace(int level){

if(level>n){

update();

}

else{

for(int i=level;i<=n;i++){

swap(x[level],x[i]);

backtrace(level+1);

swap(x[level],x[i]);

}

}

}

int main()

{

cin >> n;

memset(p,0,sizeof(p));

memset(q,0,sizeof(q));

memset(best,0,sizeof(best));

memset(x,0,sizeof(x));

for(int i=1;i<=n;i++){

for(int j=1;j<=n;j++){

cin >> p[i][j];

}

}

for(int i=1;i<=n;i++){

for(int j=1;j<=n;j++){

cin >> q[i][j];

}

}

for(int i=1;i<=n;i++){

x[i]=i;

}

backtrace(1);

cout << answer << endl;

for(int i=1;i<=n;i++){

cout << best[i]<< " ";

}

return 0;

}