第五次作业

一、算法设计题

5-2 最小长度电路板排列问题

问题表达

由于此问题为求所给定N块电路板的最佳排列，即解空间为排列树。令解向量为$\vec{x}=(x_1,x_2,\dots,x_n)$，$x_i\in\{1,2,3,\dots,n\}$，$\vec{x}$为$\{1,2,3,\dots,n\}$的一个排列。对于一个特定排列$\vec{x}$及连接块$N_k$，设$f_k(\vec{x})=\max_{x_i,x_j\in N_k}\{|i-j|\}$为此时连接块的最大长度，则搜索目标为

$$\min_{\vec{x}}\{\max_k\{f_k(\vec{x})\}\}$$

剪枝条件

此问题可行性约束函数为$x_i\in\{1,\ 2,\ 3,\ \dots,n\}$，对于限界函数，即使得$m$个连接块中最大长度达到最小，当搜索到$x_i$时，$\forall k\in\{1,2,3,\dots,m\}$其下界函数估计为$f_k(x)=\max_{1\le m,n\le i,x_m,x_n\in N_k}\{|m-n|\}$，即只考虑前$i$个搜索到的排列，未搜索到的最大长度设为0。对于当前已知的最优解$minLen$，如果$\forall k\in\{1,2,3,\dots,m\},f_k(x)<minLen$则继续搜索，若无法满足条件则剪枝。

时间复杂度

由于解集空间为排列树，故有$n!$个子问题，对于下界函数的估计需要遍历各个连接块在前x_i的连接信息，需要时间复杂度为$O(mn)$，即总时间复杂度为$O(m(n+1)!)$。

伪代码

```
int p[MAX][MAX];//p[i][j]表示第j个连接块是否连接电路板i
int minLen = INF;
int low[MAX];  //记录电路块中最左端电路板下标
int heigh[MAX];  //记录电路块中最右端电路板下标
int len(int n)
{
    int i;
    for(i=1; i<=m; i++)
    {
        low[i] = INF;
        heigh[i] = 0;
    }

    for(i=1; i<=n; i++)  //电路板
        for(int j=1; j<=m; j++)  //连接块
            if(p[x[i]][j]>0)  //如果电路板x[i]在连接块j中
            {
                if(i < low[j])
                    low[j] = i;
                if(i > heigh[j])
                    heigh[j] = i;
            }

    int max = 0;  //电路板最大长度
    for(i=1; i<=m; i++)
```

```
    if(heigh[i]>0 && low[i]<INF && heigh[i]-low[i]>max)
        max = heigh[i] - low[i];

    return max;
}
void backtrack(int i)
{
    if(i==n)
    {
        int ld = len(i);
        if(ld < minLen)
        {
            minLen = ld;
            for(int j=1; j<=n; j++)
                bestx[j] = x[j];
        }
    }
    else
        for(int r=i; r<=n; r++)
        {
            swap(x[i], x[r]);
            if(len(i) < minLen)//剪枝条件判断
                backtrack(i+1);
            swap(x[i], x[r]);
        }
}
```

5-6 无和集问题

问题表达

此问题为固定无和子集个数n搜索最大n可分的k值。对于一个特定的k，由题知集合$\{1,2,3,...,k\}$是n可分的。令解向量$\vec{x}=(x_1,x_2,x_3...,x_n)$表示为$n$个无和集，其中$x_i$表示第$i$个无和集。对于1到$k$的正整数，要将每一个元素无重复地分配到$n$个子集，即解空间为子集树。

剪枝条件

可行性约束

对于搜索到当前节点的第i个节点，表示将第i个节点的深度d放入x_i子集中，如果x_i存在两个元素相加为d，则要进行剪枝。

边界约束

对于当前节点的子节点深度为$d+1$，如果遍历所有子节点均无法将$d+1$加入，即$d+1$无法构成n可分的无和子集，则$F(n)$找到，值为d，停止搜索。

时间复杂度

由于此解空间为子集树，对于每个深度均有n个节点，共有深度为k，即最大节点数有k^n个，由于判断可行性约束需要$O(k^2)$，即总时间复杂度为$O(k^{(n+2)})$。

伪代码

```
int F[N][N],answer[N][N];
int n,maxValue;

int judge(int t,int k){
    int i,j;
```

```
    for(i=1;i<=F[k][0];i++){
        for(j=i+1;j<=F[k][0];j++){
            if(F[k][i]+F[k][j]==t)
               return 0;
        }
    }
    return 1;
}

void search(int t){
    int i,j;
    if(t>maxValue){//边界约束
        for(i=0;i<n;i++){
            for(j=0;j<=F[i][0];j++){
                answer[i][j] = F[i][j];
            }
        }
        maxValue = t;
    }

    for(i=0;i<n;i++){
        F[i][F[i][0]+1]=t;
        if(judge(t,i)){//可行性约束
            F[i][0]+=1;
            search(t+1);
            F[i][0]-=1;
        }
    }
}
```

5-16 工作分配问题

问题表达

此问题解空间为排列树，设解向量$\vec{x}=(x_1,x_2,\ldots,n)$表示第$i$个工作被分配给$x_i$工人，$x_i\in\{1,2,3,\ldots,n\}$。

剪枝条件

可行性约束

$x_i\in\{1,2,3,\ldots,n\}$。

边界约束

假设分配到第k个工作，则已知花费$C_k=\sum_i c[i][x_i]$，对于已知最优花费C_{best}，如果$C_k>C_{best}$，则需要剪枝，如果搜索到第n个工作，则需将结果返回。

时间复杂度

由于子问题共有$n!$个，每次计算需要$O(n)$，即最差时间复杂度为$O(n*n!)$。

伪代码

```
int compute(int k)//计算前k个花费
{
    int temp=0,i;
    for(i=1;i<=k;i++)
        temp+=table[i][r[i]];
```

```
    return temp;
}

void search(int k)
{
    if(k==n)//叶节点
    {
        int temp=compute(n);
        if(temp<best)
            best=temp;
        return;
    }
    for(int i=k;i<=n;i++)
    {
        swap(r[i],r[k]);
        if(compute(k) < best)//剪枝函数
            search(k+1);
        swap(r[i],r[k]);
    }
}
```

二、算法实现题

5-9 拉丁矩阵问题

问题表达

此问题求矩阵$m*n$对于n的排列，即解空间为排列树。令解向量为$a[m][n]$，$a[i][j]$表示为第i行第j列宝石形状，$a[i][j]\in\{1,2,3,\dots,n\}$。

剪枝条件

可行性约束

由于题目要求各行各列形状不同，即对于搜索到$a[x][y]$，$\exists i,m<x,j,n<y$，$a[i][j]==a[m][n]$，违反可行性条件，则需要剪枝。

边界约束

由于此题不需要对满足问题的解有要求，则只需在搜索到叶节点即可返回，停止回溯。

时间复杂度

对于解集空间最大共有$m*A_n^m$个子问题，每个子问题要判断是否满足可行性约束需要$O(n)$，即最差时间复杂度为$O(m*n*A_n^m)$。

代码

见附件，直接在附件文件夹运行$latin_maxtrix.exe$即可。