高斯消元总结

一、基础

1、方向：消成上三角矩阵，从下向上回带求出各个未知数的值。

2、**结果判断**

①**无解**：

当方程中出现(0,0,…,0,a)的形式，且 a!=0 时，说明是无解的。

②**唯一解**：

条件是 k=equ，即行阶梯阵形成了严格的上三角阵。利用回代逐一求出解集。

③**无穷解**：

条件是 k<equ,即不能形成严格的上三角形,自由变元的个数即为 equ－k。但有些题目要求判断哪些变元是不缺定的，方法如下：

首先，自由变元有 var-k 个，即不确定的变元至少有 var-k 个。我们先把所有的变元视为不确定的。在每个方程中判断不确定变元的个数，如果大于 1 个，则该方程无法求解。如果只有 1 个变元，那么该变元即可求出，即为确定变元。

对于**异或方程组**来说，还会涉及如下问题：

解的个数：$2^{(equ-k)}$

解的和最小：从最后一个有变元的方程开始，如果这个方程只有一个变元，那么就计算出来；如果有多个变元，就从当前没有解出的最后

一个变元（最靠右的列）开始枚举，直到当前行不存在未知数，然后跳转到上一行进行相同操作。

3、代码实现

（一般高斯消元，如果所有方程的解都同于 mod，则依旧可以高斯消元，只是在回带的时候利用**拓展欧几里得**求出符合条件的解即可）

```
inline void uniqueans(int var)
{
    for(int i=var-1;i>=0;i--)
    {
        int tmp=g[i][var];
        for(int j=i+1;j<var;j++)
            tmp-=g[i][j]*x[j];
        x[i]=tmp/g[i][i];
    }
}

inline void gauss(int equ,int var)
{
    int col,k,maxr;
    for(k=0,col=0;k<equ&&col<var;k++,col++)
    {
        maxr=k;
        for(int i=k+1;i<equ;i++)
            if(abs(g[maxr][col])<abs(g[i][col])) maxr=i;
        if(maxr!=k)
        {
            for(int i=col;i<=var;i++)
                swap(g[k][i],g[maxr][i]);
        }
        if(g[k][col]==0) {k--;continue;}
        for(int i=k+1;i<equ;i++)
            if(g[i][col]!=0)
            {
                int lcm=getlcm(abs(g[k][col]),abs(g[i][col]));//利用 lcm 可以在求正数解时消除误差
                int ta=lcm/g[i][col],tb=lcm/g[k][col];
                for(int j=col;j<=var;j++)
                    g[i][j]=g[i][j]*ta-g[k][j]*tb;
            }
```

```
    }
    for(int i=k;i<equ;i++)
        if(g[i][var]!=0) {puts("Inconsistent data.");return;}//无解
    if(k<var) {puts("Multiple solutions.");return;}//无穷解
    uniqueans(var);//唯一解
    for(int i=0;i<m-1;i++) printf("%d ",x[i]);
    printf("%d\n",x[m-1]);
}
```

（使异或方程组的解的和最小）

```
inline void dfs(int k,int var)
{
    if(var==-1&&k==-1)
    {
        int sum=0;
        for(int i=0;i<n;i++) sum+=x[i];
        ans=min(ans,sum);
        return;
    }
    if(num[k]==var)
    {
        x[var]=g[k][n];
        for(int i=n-1;i>var;i--) x[var]^=x[i]&g[k][i];
        dfs(k-1,var-1);
    }
    else
    {
        x[var]=1; dfs(k,var-1);
        x[var]=0; dfs(k,var-1);
    }
}

inline void uniqueans(int var)
{
    for(int i=var-1;i>=0;i--)
    {
        x[i]=g[i][var];
        for(int j=i+1;j<var;j++) x[i]^=x[j]&g[i][j];
    }
    ans=0;
    for(int i=0;i<var;i++) ans+=x[i];
}
```

```
inline void gauss(int equ,int var)
{
    int k,col,maxr;
    for(k=0,col=0;k<equ&&col<var;k++,col++)
    {
        maxr=k;
        for(int i=k+1;i<equ;i++)
        {
            if(g[maxr][col]<g[i][col]) maxr=i;
            if(g[maxr][col]==1) break;
        }
        if(maxr!=k)
        {
            for(int i=col;i<=var;i++)
                swap(g[maxr][i],g[k][i]);
        }
        num[k]=col;//记录当前行最靠左的未知数
        if(g[k][col]==0) {k--;continue;}
        for(int i=k+1;i<equ;i++)
            if(g[i][col]!=0)
            {
                for(int j=col;j<=var;j++)
                    g[i][j]^=g[k][j];
            }
    }
    for(int i=k;i<equ;i++)
        if(g[i][var]!=0) {puts("impossible");return;}//无解
    if(k==var) uniqueans(var);//唯一解
    else dfs(k-1,var-1);//枚举变元
    printf("%d\n",ans);
}
```

二、高斯消元解 XOR 方程组

1、资料：莫涛_高斯消元解 XOR 方程组

2、方向：

考虑系数矩阵，每行是一个方程，每列是一个未知数在各个方程中的系数（将第 i 行的方程的所有系数状压到一个数 a[i]里）。

我们分析一下消元之后各个方程系数的状况：

由于我们每次选择最大的一个a[i]，并且找到它最高位上的1，把其它**所有方程**（包含当前行以上的方程）这一位的系数全部消去，也就是说对于每个方程，它的系数a[i]最高位上的1所在的那一列，仅有这一个1，其余的都是0。

再进一步，如果方程个数n足够多的话，那么消元之后系数矩阵的每一行仅有一个1，并且这个1所在的那一列也仅有这一个1。

3、**经典例题**（很多其他类型的题都可以转化为经典例题）

一个性质：消元前k个a[i]异或和都能有消元后的p个a[i]的异或和组成

①从N个数中选出任意个数，使XOR和最大

解：这n个数看做二进制数，第i个数的每一位都作为第i个方程的系数（也就是上面的a数组），然后用上面的方法高斯消元，最后把消元之后的a数组全都xor起来就是答案。

②给n个数，选出若干个，使这些数还有m的xor和最大

解：先做一遍①中的算法，然后m的哪一位上不是1，就把这一位是1的那个ai异或上去即可

③(Clover 9 T3 Xor&Sum)给n个数，每次可以选择两个数ai、aj，用ai xor aj来替换ai，问最后能得到的最大的∑ai是多少

解：先做一遍题目一的算法，得到了一个能xor出的最大值。然后用这个值分别去xor a2~an，把结果累加，最后再加上这个最大值即可。

④(WC2011 xor)把问题扩展到图上——求一条1~N的路径使得路径上各条边的异或值最大

解：这样的一条路径可以看做一条1~N的简单路径加上若干的独立的环（因为可以从任何一个点走到一个独立环再走回去，相当于只异或上了一个独立环的值）。那么可以bfs求出任意一条1~N的简单路径的异或和，并且处理出所有独立环的异或值。找环的做法是：如果对于一条边(u,v,w)，dis[u] xor dis[v] xor w!=0，那么这就是一个独立环的值了。然后把这条简单路径看做m、独立环看做n个数，转化为②。

(Ps:以例子题摘自 lydrainbowcat 的博客)

⑤(HDU 3949)给你N个数，要你从中取出若干个进行异或运算，求最后所有可以得到的异或结果中的第k小的异或值

解：将这n个数进行消元后，假设有r行不全为0，则能够得到2^r个不同的**正整数**（消元后的每个选或不选），然后根据k的每个二进制位确定消元后的每个数选或不选，所有选的数的异或和就是答案。注意r<n时，说明可以异或出0，会对答案产生影响，注意特判。

⑥(BZOJ 2844)N个正整数，从中选出若干个数(或者不选)进行xor，可以得到2^N个数(不去重)。将这2^N个数从小到大排成一列。给出一个数W，问数W在这列数中第一次出现的位置。

解：和⑤很像。将这n个数进行消元后，假设有r行不全为0，n-r行全为0。前文提到可以得到2^r个不同的**正整数**，对于这2^r个**正整数**共有$2^{(n-r)}$中方法凑出（每个全为0的行选或不选），然后算出W在这2^r中的排名即可。

4、代码实现

（我习惯的命名为拓展高斯消元 exgauss）

```
inline void exgauss(int equ,int var)
{
    for(int i=1,maxr;i<=equ;i++)
    {
        maxr=i;
        for(int j=i+1;j<=equ;j++)
            if(a[j]>a[maxr]) maxr=j;
        swap(a[i],a[maxr]);
        if(a[i]==0) break;
        for(int j=var;j>=0;j--)
            if(a[i]&fac[j])
            {
                for(int k=1;k<=equ;k++)//从第一行开始消
                    if(i!=k&&(a[k]&fac[j])) a[k]^=a[i];
                break;//!!
            }
    }
}
```