2-SAT

SAT指适定性问题，k-SAT表示k-适定性问题。

可以证明k>2时，k-SAT是NP完全的，所以一般讨论k=2的情况。

思想

对于2-SAT问题，本质上是对NP问题的暴力求解。

（1）我们可以以每一个变量及其反变量为点进行建图。若x1&x2=0,则连边方式为x1->!x2,x2->!x1，简单来说，如果两变量矛盾，就各自与另一变量的反变量连边，这是非常明确的。

（2）连边之后，有无解的条件就转化为了判断是否有环，因此跑一边tarjan进行缩点，然后逐个判断变量及其反变量是否在统一强连通分量即可。

（3）在有解的情况下，两变量是由前后推导的关系的，这在拓扑序(dfn)上的表现为：我们要取的是拓扑序较大的那个值，这在tarjan的过程中求的是反序。

步骤

（1）建图

（2）Tarjan缩点=>DAG

（3）判断合法性=>对称及传递

（3）拓扑排序=>反序

例题

HDU3062Party

```
#include<bits/stdc++.h>
using namespace std;
const int maxn=2005;

int n,m,a1,a2,b1,b2,vis[maxn];
int idx,dfn[maxn],low[maxn],scc[maxn],sccnum;
int stk[maxn],top;
```

```cpp
vector<int>G[maxn];
inline void tarjan(int x){
    dfn[x]=low[x]=++idx;
    stk[++top]=x;
    vis[x]=1;
    for(int i=0;i<G[x].size();i++){
        int to=G[x][i];
        if(!dfn[to]){
            tarjan(to);
            low[x]=min(low[x],low[to]);
        }else if(vis[to]) low[x]=min(low[x],dfn[to]);
    }
    if(dfn[x]==low[x]){
        int y;++sccnum;
        while(y=stk[top--]){
            scc[y]=sccnum;
            vis[y]=0;
            if(x==y) break;
        }
    }
}

inline void add(int x,int y){
    G[x].push_back(y);
}

bool check(){
    for(int i=0;i<n;i++){
        if(!dfn[i]) tarjan(i);
    }
    for(int i=0;i<n;i+=2){
        if(scc[i]==scc[i+1]) return false;
    }
    return true;
}

void init(){
    idx=0;
    sccnum=0;
    top=0;
    for(int i=0;i<n;i++) G[i].clear();
    memset(low,0,sizeof(low));
    memset(dfn,0,sizeof(dfn));
    memset(scc,0,sizeof(scc));
    memset(vis,0,sizeof(vis));
}

signed main(){
    ios::sync_with_stdio(0);
    cin.tie(0);cout.tie(0);
    while(cin>>n>>m){
        n<<=1;
        init();
        for(int i=1;i<=m;i++){
            cin>>a1>>b1>>a2>>b2;
            add(2*a1+a2,2*b1-b2+1);
            add(2*b1+b2,2*a1-a2+1);
        }
```

```
        if(check()) puts("YES");
        else puts("NO");
    }
    return 0;
}
```

luoguP4782 【模板】2-SAT 问题

有n个向量$x_1 \sim x_n$，有m个关系，每个关系i, a, j, b规定两个变量须满足x_i为a或x_j为b，其中$a, b \in \{0, 1\}$，输出是否有解，并输出构造解。

```
#include<bits/stdc++.h>
using namespace std;
const int maxn=2e6+7;

int read(){ int x=0,f=1;char ch=getchar();while(ch<'0'||ch>'9'){if(ch=='-')
f=f*-1;ch=getchar();}while(ch>='0'&&ch<='9'){x=x*10+ch-'0';ch=getchar();}return
x*f;}

int n,m;
int dfn[maxn],low[maxn],idx;
int stk[maxn],top,scc[maxn],sccnum;
bool vis[maxn];
vector<int>G[maxn];

inline void tarjan(int x){
    dfn[x]=low[x]=++idx;
    stk[++top]=x;
    vis[x]=1;
    for(int i=0;i<G[x].size();i++){
        int to=G[x][i];
        if(!dfn[to]){
            tarjan(to);
            low[x]=min(low[x],low[to]);
        }else if(vis[to]) low[x]=min(low[x],dfn[to]);
    }
    if(low[x]==dfn[x]){
        int y;
        sccnum++;
        while(y=stk[top--]){
            scc[y]=sccnum;
            vis[y]=0;
            if(x==y) break;
        }
    }
}

inline void add(int x,int y){
    G[x].push_back(y);
}

bool solve(){
    for(int i=1;i<=2*n;i++){
        if(!dfn[i]) tarjan(i);
    }
```

```
    for(int i=1;i<=n;i++){
        if(scc[i]==scc[i+n]) return 0;
    }
    return 1;
}

signed main(){
    n=read();m=read();
    int x,y,a,b;
    for(int i=1;i<=m;i++){
        x=read();a=read();y=read();b=read();
        int aa=a^1,bb=b^1;
        add(x+n*aa,y+n*b);
        add(y+n*bb,x+n*a);
    }
    if(!solve()){
        printf("IMPOSSIBLE\n");
        return 0;
    }else{
        puts("POSSIBLE");
        for(int i=1;i<=n;i++){ //强连通分量小->拓扑序大->较优解
            if(scc[i]>scc[i+n]) printf("1 ");
            else printf("0 ");
        }
    }
    return 0;
}
```

CF776D The Door Problem

有 n扇门和 m 把钥匙，初始时有些门开着，有些门关着。一把钥匙对应多个门，每个门对应恰好两把钥匙。使用一把钥匙会使该钥匙对应的所有门的状态改变（开变成关，关变成开），问是否有存在一种钥匙的使用方案使得所有门都被打开。

扩展域并查集/2-SAT。对于开着的门，将$k1, k2$并起来，$k1+m, k2+m$并起来；对于关着的门，将$k1+m, k2$并起来，$k1, k2+m$并起来。如果存在$find(i)==find(i+m)$，则无解。

```
#include<bits/stdc++.h>
using namespace std;
const int N=2e5+7;

int n,m,a[N],fa[N];
int G[N][3],cnt[N];

int find(int x){
    if(x==fa[x]) return x;
    return fa[x]=find(fa[x]);
}

void merge(int x,int y){
    x=find(x),y=find(y);
    if(x!=y) fa[x]=y;
}

signed main(){
```

```
    ios::sync_with_stdio(0);
    cin.tie(0);cout.tie(0);
    cin>>n>>m;
    for(int i=1;i<=2*m;++i) fa[i]=i;
    for(int i=1;i<=n;++i) cin>>a[i];
    for(int i=1,k;i<=m;++i){
        cin>>k;
        for(int j=1,to;j<=k;++j) cin>>to,G[to][cnt[to]++]=i;
    }
    for(int i=1;i<=n;++i){
        if(!a[i]){
            merge(G[i][0],G[i][1]+m);
            merge(G[i][1],G[i][0]+m);
        }else{
            merge(G[i][0],G[i][1]);
            merge(G[i][0]+m,G[i][1]+m);
        }
    }
    for(int i=1;i<=m;++i) if(find(i)==find(i+m)){cout<<"NO\n";return 0;}
    cout<<"YES\n";
    return 0;
}
```

参考资料

https://www.cnblogs.com/cjjsb/p/9771868.html

https://www.cnblogs.com/xcg123/p/11818059.html