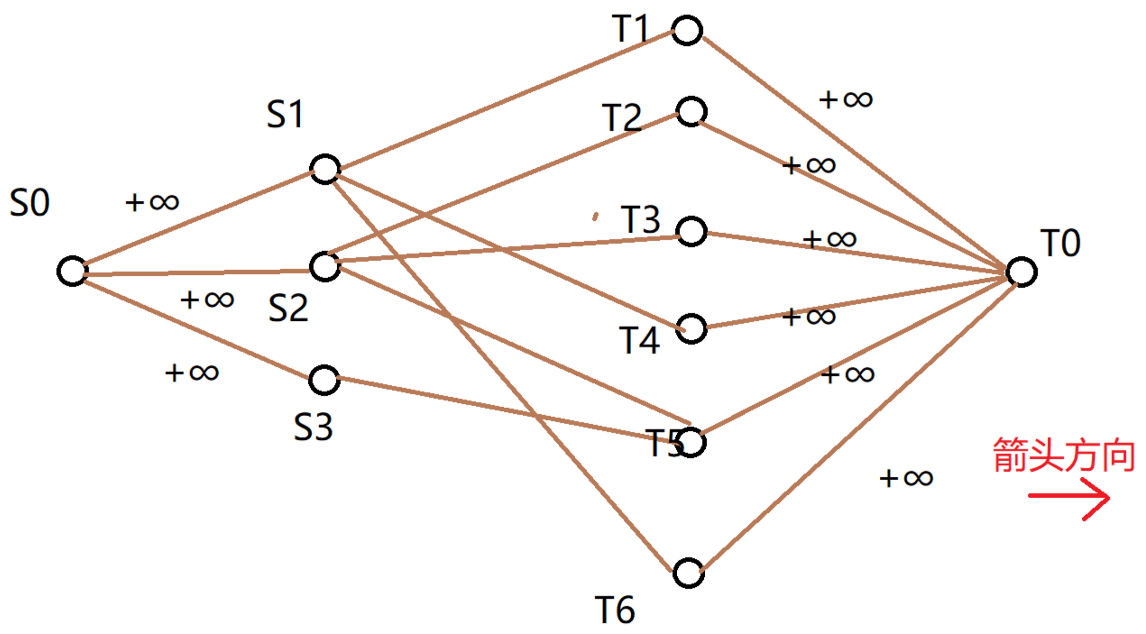
网络流 _5 多源汇最大流

多源汇最大流问题

> 给定一个n个点m条边的有向图，给定每条边的容量，非负；其中存在多个(`Sc`个)源点和多个(`Tc`个)汇点，求网络的最大流

思路：建立虚拟源点S0和虚拟汇点T0,从S0向所有源点Si连接一条容量为`INF`的边，从所有汇点Ti向汇点T0连接一条容量为`INF`的边；最后从S0向T0跑一遍最大流，既是最后的答案（是不是很简单~）

- 证明：



> 对于中间所有边都满足容量限制和流量守恒，由于左右两边的所有边的容量都是正无穷，所以必然满足容量限制。因此对于构造的图满足原图一个可行流对应新图的一个可行流。
>
> 将S0流出和流入T0的边全部删去，仍然中间的点组成一个流网络，因此新图的仍以一个可行流都能对应原图的一个可行流。证毕。

`代码`：

```
const int N = 10010 , M = (N + 100010) * 2 , INF = 1e8;
int n,m,S,T;
int e[M],ne[M],h[N],f[M],idx=0;
int q[M],d[N],cur[N];

void add(int a,int b,int c){
    e[idx] = b , f[idx] = c , ne[idx] = h[a] , h[a] = idx ++;
    e[idx] = a , f[idx] = 0 , ne[idx] = h[b] , h[b] = idx ++;
}

int find(int u,int limit){
    if(u == T) return limit;
    int flow = 0;

    for(int i=h[u];~i && flow < limit ; i = ne[i]){
        cur[u] = i;
```

```
        int ver = e[i];
        if(d[ver] == d[u] + 1 && f[i]){
            int t = find(ver,min(f[i],limit - flow));
            if(!t) d[ver] = -1;
            f[i] -= t , f[i^1] += t , flow += t;
        }
    }
    return flow;
}

bool bfs(){
    memset(d,-1,sizeof d);
    int hh = 0, tt = 0;
    q[0] = S , cur[S] = h[S] , d[S] = 0;

    while( hh <= tt ){
        int u = q[hh ++];
        for(int i = h[u] ; ~i ; i = ne[i]){
            int ver = e[i];
            if(d[ver] == -1 && f[i]){
                d[ver] = d[u] + 1;
                cur[ver] = h[ver];
                if(ver == T) return true;
                q[++ tt] = ver;
            }
        }
    }
    return false;
}

int dinic(){
    int ans = 0 , flow = 0;
    while(bfs()) while(flow = find(S,INF)) ans += flow;
    return ans;
}
//=================================
int main(){
    memset(h,-1,sizeof h);

    int sc = 0 ,sd = 0, s = 0 , t = 0;
    n = read() ,m = read() , sc = read() , sd = read();
    S = 0 , T = n + 1;
    while(sc -- ){
        s = read();
        add(S,s,INF);
    }
    while(sd --){
        t = read();
        add(t,T,INF);
    }

    rep(i,1,m){
        int a = read() , b = read() , c = read();
        add(a,b,c);
    }
    print(dinic());
    return 0;
```

```
}
```