图论

一、支配

1. 支配集

无向图：$G=<V,E>$

支配集：$V^* \subseteq V$使得$\forall u \in V-V^*, \exists v \in V^*, uEv$

一个顶点子集，使得不属于这个子集的顶点都与子集内的某顶点相邻。

极小支配集：V^*是支配集，其真子集都不是

最小支配集：$|V^*|$最小的支配集

支配数：$\gamma_0(G)=|V^*|$，且V^*是最小支配集

2. 定理

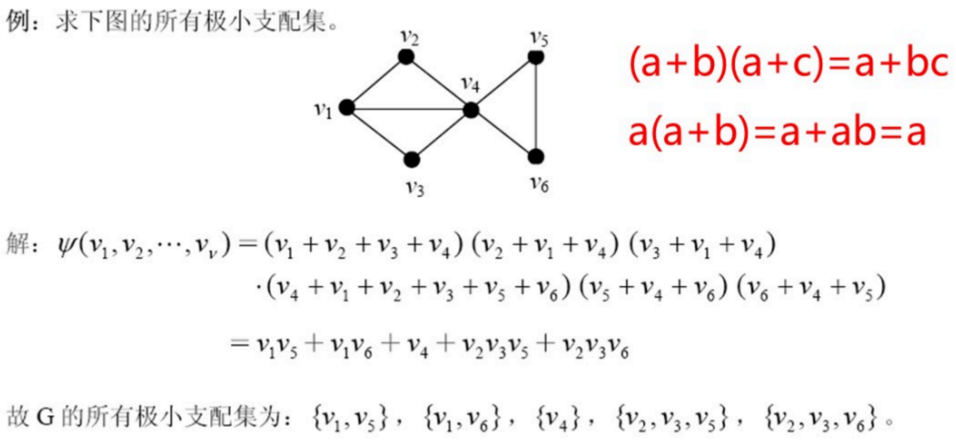
无向图G无孤立点，V_1^*是极小支配集，则存在V_2^*也是极小支配集，且$V_1^* \cap V_2^* = \varnothing$

3. 求解最（极）小支配集

$$\psi(v_1,v_2,\cdots,v_n)=\prod_{i=1}^{n}(v_i+\sum_{u\in N(v_i)}u)$$

$N(v)$为v的邻域，求左式的最简化析取式，每一项对应一个极小支配集。

> (a + b)(a + c) = a + bc
>
> a(a + b) = a + ab = a



二、覆盖

1. 点覆盖

无向图$G=<V,E>$

点覆盖：$V^* \subseteq V$使得$\forall e \in E, \exists v \in V^*$，v关联e

一个顶点的子集，使得所有的边都与子集中的某顶点关联。

极小点覆盖：V^*是点覆盖，其真子集都不是

最小点覆盖：$|V^*|$最小的点覆盖

点覆盖数：$\alpha_0(G)=|V^*|$，且V^*是最小点覆盖

2. 定理

连通图中，点覆盖一定是支配集。

极小点覆盖不一定是极小支配集。

支配集不一定是点覆盖。

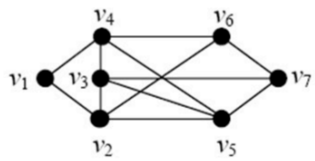
3. 求解最（极）小点覆盖

$$\psi(v_1,v_2,\cdots,v_n)=\prod_{i=1}^{n}(v_i+\prod_{u\in N(v_i)}u)$$

求最简析取式。

(a+b)(a+c)=a+bc

a(a+b)=a+ab=a



解：$\varphi(v_1,v_2,\cdots,v_7)=(v_1+v_2v_4)(v_2+v_1v_3v_5v_6)(v_3+v_2v_4v_5v_7)(v_4+v_1v_3v_5v_6)$
$\cdot(v_5+v_2v_3v_4v_7)(v_6+v_2v_4v_7)(v_7+v_3v_5v_6)$

$=v_1v_3v_5v_6+v_2v_3v_4v_5v_6+v_2v_4v_5v_7+v_2v_3v_4v_7$。

故极小点覆盖有：

$C_1=\{v_1,v_3,v_5,v_6\}$，$C_2=\{v_2,v_3,v_4,v_5,v_6\}$，$C_3=\{v_2,v_4,v_5,v_7\}$，$C_4=\{v_2,v_3,v_4,v_7\}$。

*4. 边覆盖

无向图$G=<V,E>$

边覆盖：$E^*\subseteq E$，使得$\forall v\in E,\exists e\in E^*$，e关联v

所有边的一个子集，任意顶点均与子集中的某条边关联

（一个能包含所有事物的关系集合）

极小边覆盖：E^*是边覆盖，其真子集都不是

最小边覆盖：$|E^*|$最小的边覆盖

边覆盖数：$\alpha_1(G)=|E^*|$，且E^*是最小点覆盖

三、独立

1. 独立集

无向图$G=<V,E>$

独立集：$V^*\subseteq V$使得$\forall u,v\in V^*,(u,v)\notin E$

一个顶点的子集，其中任意两个顶点之间没有边相连

（根据给定关系，找出一个互不干扰的事物的集合）

极大独立集：V^*是边覆盖，其真母集都不是

最大独立集：$|V^*|$最大的独立集

独立数：$\beta_0(G)=|V^*|$，且V^*是最大独立集

2. 定理

无向图G无孤立点，V^*是极大独立集，则V^*也是极小支配集

***极小支配集不一定是极大独立集

四、支配、覆盖、独立总结

1. 团

无向图$G=<V,E>$，$V^*\subseteq V$

团：$G[V^*]$是完全子图

极大团：V^*是团，其真母集都不是

最大团：$|V^*|$最大的团

团数：$v_0(G)=|V^*|,V^*$是最大团

2. 定理

无向图G无孤立点，$V^*\subset V$，V^*是点覆盖 $\Longleftrightarrow V-V^*$是独立集

3. 总结

- V_1^*是极小支配集，则存在V_2^*也是极小支配集，且$V_1^*\cap V_2^*=\varnothing$
- V^*是点覆盖，则V^*也是支配集
- V^*是极大独立集，则V^*也是极小支配集
- V^*是点覆盖 $\Longleftrightarrow V-V^*$是独立集
- V^*是G的团 $\Longleftrightarrow V^*$是$G_{补}$的独立集

四、匹配

1. 匹配

无向图$G=<V,E>$

匹配（边独立集）：$E^*\subseteq E$使得$\forall e,f\in E^*$，e和f不相邻

边的一个子集，子集中任意两条边不相邻（顶点不重合）（若干对不同事物之间的二元关系）

极大匹配：E^*是匹配，其真母集都不是

最大匹配：$|E^*|$最大的匹配

匹配数：$\beta_1(G)=|E^*|$，其中$|E^*|$是最大匹配

2. 饱和点，交错路径，增广路径

设M是G中的匹配

- 饱和点：v与M中边关联
- 非饱和点：v不与M中边关联

- 交错路径：在M和E-M中交替取边的路径
- 可增广交错路径：两端都是非饱和点的交错路径

完美匹配：没有非饱和点的匹配

3. 求解最大匹配

从一个匹配开始，逐一检查不饱和点，对每个不饱和点尝试寻找增广路径。

得到更大的匹配，递归直到没有不饱和点或没有增广路径。

```
bool find(int u){ // 寻找增广路径
    for(int i = h[u]; i != -1; i = ne[i]){
        int j = e[i];
        if(!st[j]){ // 第一次找到
            st[j] = true;
            if(!match[j] || find(match[j])){ // 不饱和点 | 匹配点能找到增广路径
                match[j] = u;
                return true;
            }
        }
    }
    return false;
}
for(int i = 1; i <= n1; i ++){ // 对每个不饱和点
    memset(st, false, sizeof st);
    if(find(i))ans ++;  // 更新匹配数
}
```

五、网络流

1. 记号

> V： 点集
>
> E： 边集
>
> G = <V, E>：图
>
> s : 源点
>
> t : 汇点
>
> c(u, v)：边<u, v>的容量
>
> f(u, v)：边<u, v>的实际流量

2. 性质

- 容量限制：$f(u,v) \leq c(u,v)$
- 反对称性：$f(u,v) = -f(v,u)$
- 流量平衡：对于不是源点也不是汇点的任意节点，$\sum\limits_{u \in V} f(v,u) = 0$

满足上述三条性质的网络称为网络流，也称为可行流，可行流至少有一个：零流。

3. 最大流问题

定义一个网络的流量$F=\sum\limits_{v\in V}f(s,v)$（从源点流出的总流量）

最大流问题：满足上述三条性质的前提下求F_{max}

4. 弧的分类

零流弧：$f(u,v)=0$

非零流弧：$f(u,v)>0$

饱和弧：$f(u,v)=c(u,v)$

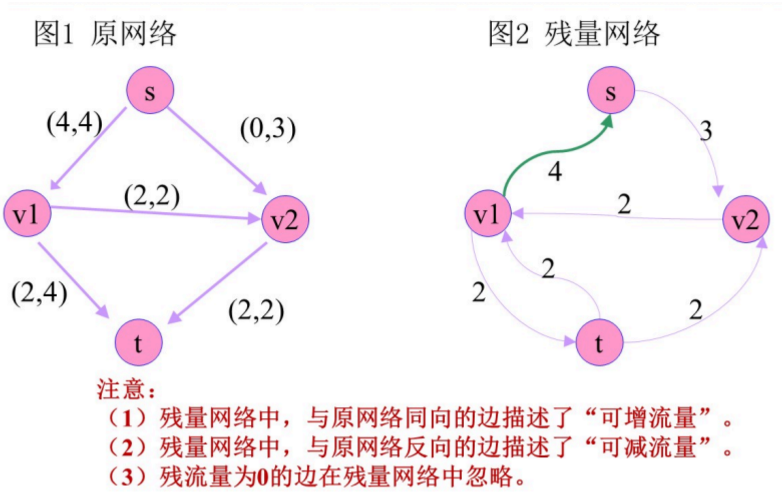
非饱和弧：$f(u,v)<c(u,v)$

前向弧：与路的方向一致的边

后向弧：与路的方向相反的边

5. 残量网络

可增流量和可减流量都要表示出来。



6. 标号法

从一个可行流F出发（可以设为零流），经过标号过程和调整过程。

标号形式：

- 网络中的顶点或者标号点（分为已检查和未检查两种），或者是未标号点
- 对于每个标号顶点V_i，其标号分为两部分：$(V_x,L(V_x))$
  - V_x表示V_i的标号从哪个顶点得到
    - 若V_x前面为$'+'$号，表示一条有向边(V_x,V_i)，且该有向边流量不满，V_x到V_i的流量可增加
    - 若V_x前面为$'-'$号，表示一条有向边(V_i,V_x)，且该有向边流量不为0，可由V_i到V_x的流量可减少
  - $L(V_x)$表示可增加或减少的最大流量

标号过程：

```python
s, t, infty = 0, n - 1, 1e9
def Signed(V, E):
    V[s].sign = (0, infty)
    q = queue()
    q.push(s)
    while q.size:
        t = q.front()
        q.pop()
        for e in V(t):  # t的所有邻居
            if(f(t, e) < c(t, e) and f(t, e) >= 0):
                e.sign = (t, min(t.sign[1], c(t, e) - f(t, e)))
            elif(f(t, e) < 0):
                e.sign = (-t, min(t.sign[1], -f(t, e)))
        if nothing_signed:
            return False
    return True
def NetworkFlow(V, E):
    while Signed(V, E) = True:
        path = find_path(V, E)  # 找到一条可调整路径
        P_plus = pos_arc(path)  # 可调整路径的前向弧集合
        P_neg = neg_arc(path)   # 可调整路径后向弧集合
        a = V[t].sign[1]        # 找到最小的L(x)，即终点流量值
        for arc in P_plus:
            arc += a    # 对所有前向弧增加a流量
        for arc in P_neg:
            arc -= a    # 对所有后向弧减少a流量
```

*7. 最大流最小割定理

割集间的容量：$C(S,T)=\sum\limits_{x\in S}\sum\limits_{y\in S}c(x,y)$

即：所有由S中的一点指向T中的一点的边上的容量之和

任意一个网络D中的从V_s到V_t的最大流的流量等于分离V_s和V_t的割集中容量最小的割集的容量。

以下三个条件等价：

> f是最大流
>
> 残量网络中找不到增广路径
>
> $|f|=C_{min}(S,T)$

运算法则：

> $f(X,X)=0$
>
> $f(X,Y)=-f(Y,X)$
>
> $f(X\cup Y,Z)=f(X,Z)+f(Y,Z)$
>
> $f(X,Y\cup Z)=f(X,Y)+f(X,Z)$

定理：

- 任意不包含s和t的顶点集X，与它相关联的边上的流量之和为0
- 整个网络的流量等于任意割的流量
- 整个网络的流量不可能超过容量最小的割的容量

构成最小割的边的求解：

> 1. 求最大流
> 2. 求最大流的残量网络
> 3. 在残量网络中，遍历所有可以由源点出发到达的节点，其集合为S，T = V - S
> 4. 所有连接S和V的边称为最小割的边

*8. 利用网络流求二分图的最大匹配

(1) 从二部图 G 出发构造一个容量网络 G′，步骤如下：

a) 增加一个源点 S 和汇点 T；

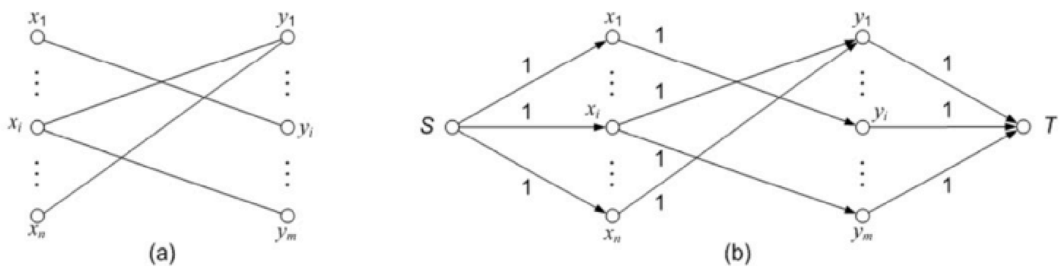
b) 从 S 向 X 的每一个顶点都画一条有向弧，从 Y 的每一个顶点都向 T 画一条有向弧；

c) 原来 G 中的边都改成有向弧，方向是从 X 的顶点指向 Y 的顶点；

d) 令所有弧的容量都等于 1。

(2) 求容量网络 G′ 的最大流 F。

(3) 最大流 F 求解完毕后，从 X 的顶点指向 Y 的顶点的弧集合中，弧流量为 1 的弧对应二部图最大匹配的边，最大流 F 的流量对应二部图的最大匹配的边数。



六、博弈论

1. 博弈的三要素

- 参与人
- 策略集
- 回报

每个参与人有一个策略集，

策略组：每个参与人出一个策略构成的策略组合

对应每个策略组，每个参与人有一个回报

2. 博弈的目标

win or double-win

3. 假设

- 能够做出最优判断
- 互相了解对方策略集
- 参与人目标是个人利益最大化

5. 博弈的计算

收益矩阵：

	抵赖	坦白
抵赖	-1， -1	-10，0
坦白	0，-10	-4，-4

6. 博弈的结果

均衡：在这个策略组下，任何参与人都没有改变的机会

7. 博弈论思想

情景描述→收益矩阵→寻求均衡

8. 最佳应对&占优策略

严格最佳应对：如果你这么做，我最好选择这样做

严格占优策略：不管你怎样做，我都最好选择这样做

对于囚徒困境："坦白"此时是疑犯1和疑犯2的"互为严格占优策略"

***至少一方存在严格占优策略，博弈结果可预测。

9. 简单博弈的行为推理

- 如果两个参与者都有严格占优策略，可预计他们都会采取严格占优策略
- 如果只有一个人有严格占优策略，则这个人会采取严格占优策略，另一方会采取此策略的最佳应对

*10. 协调博弈

如果两个参与人的策略组合分别构成各自的严格占优策略，那么这个组合就被定义为纳什均衡。

存在多个纳什均衡的博弈称为"协调博弈"