1120132001

图论选

图的基本术语（哈工大版）

- G=(V,E)
- 环(loop, self-loop)
- 重边(multiple adjacency)
- 通道(walk, chain, path)
- 闭通道(closed walk, cycle)
- 迹(trail)
- 闭迹(closed trail, tour, circuit)
- 路(path, simple path)
- 圈(cycle, simple cycle, circuit)

图的基本术语

- 简单图、无向图、伪图<->**多重图**、无向图、伪图
- 无向图、伪图<->**有向图**、图

- 业界不成文的规定，任何相关书目必须给出以上术语的定义，不然谁都不知道啥意思.....

图的基本术语

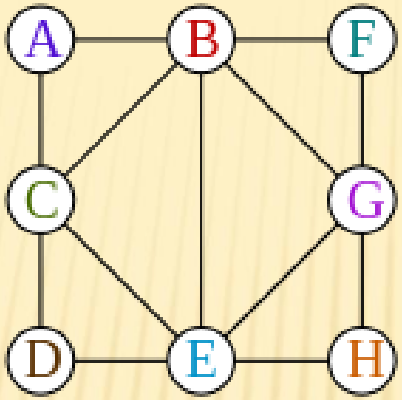
- 子图 -> 乌拉尔猜想
- 生成(spanning)

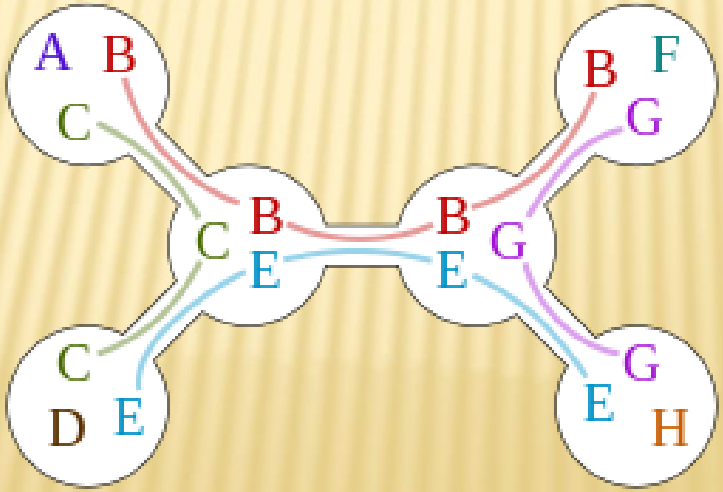
- 欧拉环游，欧拉闭迹
- 哈密顿圈

树分解(TREE DECOMPOSITION)

- 树分解
- 注意区别于生成树、树分治和树剖分(HLD)
- 构造一棵树，使得：
  - 原图的每个顶点在至少一个树顶点中
  - 原图的相邻顶点至少一次出现在同一个树顶点中
  - 不邻接的树顶点一定不包括共同的原图顶点
- 树宽：最大树顶点的大小-1
- 图的树宽：所有树分解树宽的最小值

树分解(TREE DECOMPOSITION)





DAG图多DP——ACDREAM 1083

- 甲要从0号点尽快走到N-1号点，城管可以选择在任何时候炸掉一条边，但是不能炸甲正在走的边。甲希望尽快到，城管希望尽量延长时间，双方都能及时知道所有的情报，求甲用时。

DAG图多DP——ACDREAM 1083

- 拓扑排序
- 然后DP

DAG图多DP——ACDREAM 1083

- DP过程中算一个最短路，一个次短路作为中间变量。
- dist[i]=min{dist[j]+w[i,j]}
- snddist[i]=min{dist[j]+w[i,j]|j!=argmin dist[i]}
- val[i]=max{snddist[i],min{val[j]+w[i,j]}}
- 答案是val[0]

最短路

- 不讲了
- 注意活用
- 理解清楚Bellman方程真正的含义，以及每一步转化的意义的话这很容易。
- 如CF257-D和POJ3635

最短路

- 第K最短路：先算每个点到终点的距离，然后A-star。这样一定会按从短到长的顺序到达终点。
- 恰好K步最短路：用Floyd-Warshall迭代。
- 平均最短路：分数规划，二分答案然后找合适的环即可。

STEINER树

- 看我的网络流PPT好了。
- 本质上就是个状压DP，不过合理地选择了压缩的方式。
- 特点：关键点的数目不超过10。

有向图最小树形图——HDU4966

- 总共有n门课，每门也有一个最高等级ai，一开始每门都是等级0。
- 有m个提升班，每个班级有a l1 b l2 w，表示只有第a门课程在l1及以上时才能上，花费代价w，可以讲第b门课程提升到l2等级。
- 求将所有的课程提升到最高等级需要付出的最小代价。

有向图最小树形图——HDU4966

- 所有课程的每个等级视为一个节点。
- 对于每门课程的等级i,可以建一条指向等级i-1的有向边，边权为0。
- 对于每个提升班，可以建一条边权为w的从(a,l1)到(b,l2)的有向边。
- 求最小树形图（朱刘算法）。

朱刘算法

- 全是板题，算法怎么跑我也忘了，百度吧。
- 有板就行，不用深究。
- 关键要会构图。（小心和网络流混淆）

HDU5036

- N个带锁房间，每个房间对应一种钥匙。
- 每个房间里随机放一个或多个钥匙。
- 当没有门可打开的时候就炸开一个随机房间。
- 求一共炸开房间的期望数。

HDU5036

- 构成一个有向图。
- 每个顶点可以被能到达它的任何顶点解锁。炸开的概率是顶点数分之一。
- 用Warshall算法
- 注意可达矩阵常用位图优化

欧拉回路

- 很神奇的算法，可以解决不少奇怪的题目。
- 比如说JTY给15级出的那道。
- 已知每场比赛前的CF积分和比赛的积分变化，求曲线是否合理。

- 有无判定谁都懂，但是乱走可能会卡死的。

欧拉回路

- Euler-circuit(st)
  - 对所有边(st,ed)
    - 标记边(st,ed)。如果是有向图，也标记反向边。
    - Euler-circuit(ed)
    - (st,ed)入栈
- 最后全部出栈就可以。

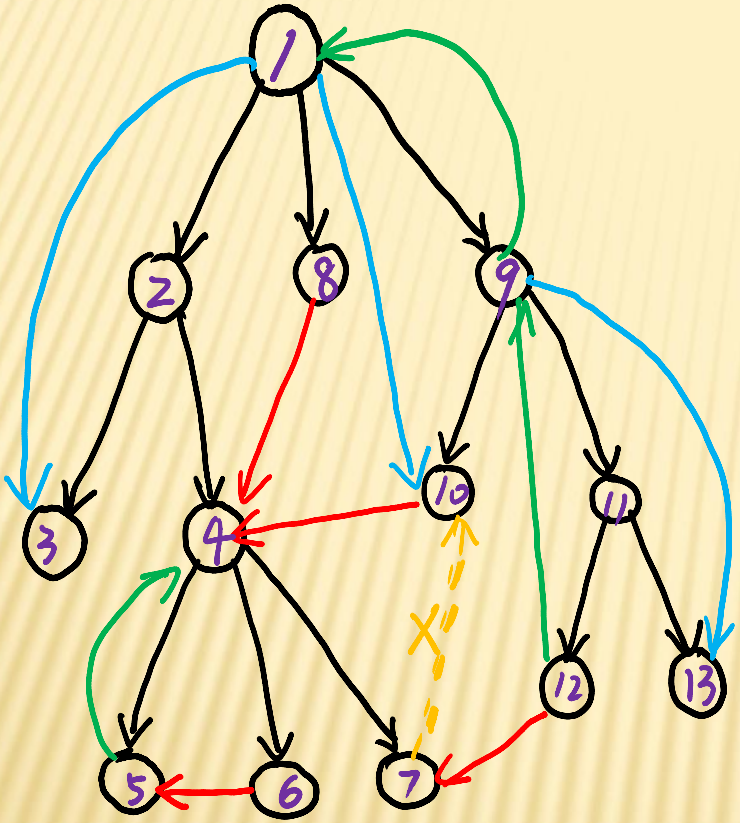
HAVEL定理

- 根据度数构造简单图。
  - 1.按度数排序
  - 2.选择最大的顶点v
  - 3.v按从大到小的顺序依次和其他顶点连边，直到v的度数用完。如果无法满足，则问题无解。
  - 4.如果还有顶点度数不为0，回到1重复。
- 就是个贪心，证明就不给了。
- 注意构图通常都不是唯一解。

无向图全局最小割

- 拿网络流需要枚举顶点，复杂度太高。
- 用Stoer-Wagner算法。
- 有点像Prim最小生成树，有板就行，只能出板题，考的概率非常低。

TARJAN算法



→ 树枝边

→ 横叉边

→ 前向边

→ 后向边

12 时间

-x→ 不可能边

TARJAN算法

- DFN：时间戳(DFS Number)

- 设当前的DFS节点为x，下一个节点是y
  - 树枝边：DFN[y]不存在
  - 后向边：DFN[y]<DFN[x]，y在栈中（额外标记）
  - 横叉边：DFN[y]<DFN[x]，y不在栈中
  - 前向边：DFN[y]>DFN[x]

- 思考：无向图有几种边?

TARJAN算法

- 无向图
  - 横叉边->树枝边
  - 前向边->后向边
  - 因此只有树枝边和后向边

TARJAN算法

- low'：X所能到达的点的最小DFN
- 如果low'[x]=DFN[x]则x是强成分的根

- 这可能实现吗?
- 需要多遍迭代DFS，计算不便.....

- 弱化条件

TARJAN算法

- low：X经过任意数量的树枝边，和至多一条非树枝边所能到达的点的最小DFN
- low[x]=min{DFN[x], DFN[y], low[z]}
- 式中y在x的子树中，z不在x的子树中

- 注意如果low'[x]<DFN[x]，那么一定有low[x]<DFN[x]，所以没关系

TARJAN算法——伪代码实现

- Tarjan-DFS(x)
  - x.low = x.DFN = ++ timestamp
  - foreach (x, y) in E
    - if y.DFN doesn't exist
      - Tarjan-DFS(y)
    - x.low = min(x.low, y.low)

- 和我上一页说的好像不太一样啊……
- 但是一样可以工作

TARJAN算法——伪代码实现

- **注意**：无向图一定不能用进入时的边更新low
  - 无重边的场合，拿树上的父节点判断就行
  - 有重边的场合，要用树上的父边判断
- 另：无向图没有横插边，因此DFN可以用每个节点在树中的深度代替

TARJAN算法——强成分

- Tarjan(x)开始时，首先把x加入栈
- 干原来应该的工作
- 函数返回前，判断DFN和low是否相等
- 如果相等就一直退栈直到退出x为止
- 这一轮退掉的所有节点位于同一个连通分量内

TARJAN算法——无向图割点

- 如果树的根有超过一个子树，那么删除根节点后各子树无法连通，根节点是割点
- 对于边(x, y)，如果low[y]>=DFN[x]，那么删除x后y无法从根节点到达，x是割点

TARJAN算法——无向图边双成分

- Tarjan(x)开始时，首先把x加入栈
- 出栈算法有多种，哪个都可以：
  - 对边(x, y)，若low[y]>DFN[x]，那么(x, y)是桥，出栈一直到退出y点，这些点在同一个分量内。
  - 函数返回时，若low[x]=DFN[x]，那么x的父边是桥，出栈一直到退出x点，这些点在同一个分量内。
  - 网上好像还有其他算法.....

TARJAN算法——无向图点双成分

- 每个点可能属于多个点双连通分量。
- 这里讲一个标记边的算法，用于POJ2942。
- 网上能找到标记点的算法。

TARJAN算法——无向图点双成分

- 对每一条非父边(x, y)
  - 若DFN[y]未定义
    - (x, y)入栈
    - Tarjan(y)
    - 若low[y]>=DFN[x]
      - 每一条边出栈直到(x, y)出栈，标记所有边为割点x所属的双成分内
  - 否则若DFN[y]<DFN[x]
    - (x, y)入栈
  - low[x]=min(low[x], low[y])

TARJAN算法——无向图点双成分

- 注意DFN[y]<DFN[x]这一句，有什么意义?
- 显然无向图不可能存在前向边，因此理论上这个判断是无条件成立的。

- 但是不要忘记无向图边的存储往往用邻接链表，在两个方向各存一次。
- 因此如果DFS走1->2->3，3走后向边到1，那么之后1有可能再次找到边(1, 3)

2-SAT

- 给定一堆布尔变量和一堆2元条件，求一个解。

- 算法：
  - 把每个布尔变量X变成两个点，X为真和X为假
  - 把所有条件拆成如下形式：如果A，那么B
  - 对每个形如上式的条件，从A到B连有向边
  - 强连通分量缩点，如果对所有X，X为真的点和X为假的点不在同一个成分中，问题有解
  - 黑白染色法得出一个解即可

2-SAT

- 典型连边方法：
  - x | y => 非x -> y， 非y -> x
  - x = y => 非x -> 非y，非y -> 非x
  - x -> y， y -> x
  - x ^ y =>非x -> y，非y -> x
  - x -> 非y， y -> 非x
  - x & y => 非x -> x， 非y -> y （想想为什么）
- 不要背，现场推

一个实战——HDU5409

- Tarjan算法请务必理解，学会推理。
- 一个连通无向图有N个点M条边，对每条边，判断删除它之后图是否依然联通；如果不联通输出一个点对(u,v)，要求点u和点v不联通，编号u<v且u尽可能大，v尽可能小。
- 博客的写法都太残了。如果在输入图之后只允许DFS一遍，然后直接输出答案，怎么做?

一个实战——HDU5409

- 对于每个桥，设两个连通分量中最大的顶点编号为A,B
- 如果A!=N，那么B=N，输出(A, A+1)
- 否则一定有B!=N，输出(B, B+1)

- 双连通分量缩点？
- 没必要

一个实战——HDU5409

- 考虑Tarjan算法，所有的桥一定是树枝边。
- 如何记录两边的顶点的最大编号?
- 如果保证N一定在一边，那么我们只用算另一边就好了。
- 于是我们从N号节点开始Tarjan。
- 于是要输出的u就是子树的最大编号。
- 树形DP？一并计算就好了。

一个实战——HDU5409

```
void Tarjan(NODE *x){
    x->DFN = x->low = ++ dep;
    for(EDGE *j = x->head; j; j = j->next){
        NODE *p = j->b;
        if(p == x->par)
            continue;
        if(!p->DFN){
            p->par = x;
            Tarjan(p);
            j->bridge = p->low > x->DFN;
        }
        x->low = min(x->low, p->low);
        x->mv = max(x->mv, p->mv);
        j->u = j->b->mv;
    }
}
```

DOMINATOR TREE

- 给一个有向图G，图中存在一个点r可以到达所有的点
- 如果从r走到某个点x一定经过点y，称y为x的必经点
- DFN最大的必经点成为最近必经点
- 将最近必经点关系作为父子关系构成的树，成为有向图的支配树
- x是y的必经点 ⇔ 支配树中x是y的祖先

DOMINATOR TREE

- 用于求:
- 有向图的必经关系
- 有向图的割点
- 有向图的桥

- 考的概率极低
- 例题见HDU4694和CFgym100513的L

DOMINATOR TREE

- 可以参考李煜东的PPT
- 讲得比我的清晰多了.....
- 但是他的代码是没优化过的，感觉没有我简单。

CJW找环法1——HDU5222

- 判断单向双向边混合图中是否有环
- 先用双向边缩点
- 再用单向边跑Tarjan
- 发现后向边就是环（注意要和横叉边区别）

CJW找环法2——HDU5215

- 给定一个无向图，问图中是否有长为奇数或者偶数的环。
- 标记根为红，然后开始DFS，红蓝交替染色
- 当遇到第一个奇数环时，标记节点为红蓝双色
- 染色和环那里需要分情况讨论一下

TARJAN算法跑动态图——CFGYM100551

- 这套题有很详尽的题解和代码，XYZ的课件也讲过相关问题
- 可以拿LCT做也可以那Tarjan做。这里讲讲Tarjan。

TARJAN算法跑动态图

- 核心思想是离线然后CDQ分治
- 分治的时候，每加入若干边，就缩一次点，算一次桥，然后清理掉不需要的点。
- 具体看代码吧。