图论算法选讲（金华）

diamond_duke

2019 年 10 月 2 日

定义

两个点 u, v 的**最短路**，是指在给定的一个图中的一条 u 到 v 的路径，且最小化经过的边权之和。

定义

两个点 u, v 的**最短路**，是指在给定的一个图中的一条 u 到 v 的路径，且最小化经过的边权之和。
最短路相关算法一般有如下几个（这里，我们假设 n 是图的点数，m 是图的边数）:

- Floyd 算法，用于求解图中任意两点的最短路，时间复杂度 $\Theta(n^3)$；
- Dijkstra 算法，用于求解图中某一点到其余点的最短路，时间复杂度 $\Theta(m \log_2 n)$。
- Bellman-Ford 算法，用于求解图中某一点到其余点的最短路，时间复杂度 $\Theta(nm)$。

Floyd 算法

Floyd 本质上是一个 DP 的过程：设 $dis_{k,i,j}$ 表示只使用 $1, 2, \cdots, k$ 作为中间节点时，节点 i 到 j 的最短路。

Floyd 算法

Floyd 本质上是一个 DP 的过程：设 $dis_{k,i,j}$ 表示只使用 $1, 2, \cdots, k$ 作为中间节点时，节点 i 到 j 的最短路。
考虑转移。显然 $dis_{k+1,i,j}$ 若与 $dis_{k,i,j}$ 不同，那么一定会经过 $k+1$，因此 $dis_{k+1,i,j} = \min\{dis_{k,i,j}, dis_{k,i,k+1} + dis_{k,k+1,j}\}$。

Floyd 算法

Floyd 本质上是一个 DP 的过程：设 $dis_{k,i,j}$ 表示只使用 $1, 2, \cdots, k$ 作为中间节点时，节点 i 到 j 的最短路。
考虑转移。显然 $dis_{k+1,i,j}$ 若与 $dis_{k,i,j}$ 不同，那么一定会经过 $k+1$，因此 $dis_{k+1,i,j} = \min\{dis_{k,i,j}, dis_{k,i,k+1} + dis_{k,k+1,j}\}$。
可以使用滚动数组去掉 k 的维度。
时间复杂度：$\Theta(n^3)$。

Dijkstra 算法

Dijkstra 算法的思路是每次从所有没有选择过的点当中，选择距离最小的点 u，并用 u 更新所有与它相邻的点 v 的距离。

Dijkstra 算法

Dijkstra 算法的思路是每次从所有没有选择过的点当中，选择距离最小的点 u，并用 u 更新所有与它相邻的点 v 的距离。
在**边权非负**时，u 被选择时的距离就是最终的最短路，因此 Dijkstra 只在边权非负时正确。

Dijkstra 算法

Dijkstra 算法的思路是每次从所有没有选择过的点当中，选择距离最小的点 u，并用 u 更新所有与它相邻的点 v 的距离。
在**边权非负**时，u 被选择时的距离就是最终的最短路，因此 Dijkstra 只在边权非负时正确。
可以使用堆加速选择 u 的过程，但是需要注意判断出堆的 u 是否已经被选择过了。
时间复杂度：$\Theta(m\log_2 n)$。

Bellman-Ford 算法

Bellman-Ford 算法是在 Dijkstra 算法的基础上，去掉选择距离最小的点 u，而是选择任意一个被更新过的点 u 更新其他点。

Bellman-Ford 算法

Bellman-Ford 算法是在 Dijkstra 算法的基础上，去掉选择距离最小的点 u，而是选择任意一个被更新过的点 u 更新其他点。
在最坏情况下，每条边都会更新每个节点一次，因此最坏时间复杂度为 $\Theta(nm)$。

Bellman-Ford 算法

Bellman-Ford 算法是在 Dijkstra 算法的基础上，去掉选择距离最小的点 u，而是选择任意一个被更新过的点 u 更新其他点。
在最坏情况下，每条边都会更新每个节点一次，因此最坏时间复杂度为 $\Theta(nm)$。
一个优化是不将同一个节点重复加入队列，但是并不会带来时间复杂度上的优化。

Bellman-Ford 算法

Bellman-Ford 算法是在 Dijkstra 算法的基础上，去掉选择距离最小的点 u，而是选择任意一个被更新过的点 u 更新其他点。
在最坏情况下，每条边都会更新每个节点一次，因此最坏时间复杂度为 $\Theta(nm)$。
一个优化是不将同一个节点重复加入队列，但是并不会带来时间复杂度上的优化。
Bellman-Ford 算法的优点在于不要求边权非负。

CodeForces 295B

给定 n 个点的有向图和一个排列，按照排列顺序删除有向图中的点，输出每次删除后两点间最短路之和。
$n \leq 500$。

NOI 2007 社交网络

给定 n 个点 m 条边的无向图，定义 $C_{s,t}$ 为 s 到 t 的最短路条数，$C_{s,t}(u)$ 为经过 u 的 s 到 t 的最短路条数。对于每个节点 u，求

$$I(u) = \sum_{s,t} \frac{C_{s,t}(u)}{C_{s,t}}$$

$n \leq 100$。

一道例题

给定 n 个点 m 条边的无向图，每走一步所有边的边权会从 w 变成 $\frac{1}{1-w}$，求 1 到 n 的最短路。
$1 \leq n, m \leq 5000$。

强连通分量

强连通分量是指有向图的一个子图，满足子图中所有点之间可以两两到达。

强连通分量

强连通分量是指有向图的一个子图，满足子图中所有点之间可以两两到达。我们可以用 Tarjan 算法来求出所有的极大强连通分量：

强连通分量

强连通分量是指有向图的一个子图，满足子图中所有点之间可以两两到达。我们可以用 Tarjan 算法来求出所有的极大强连通分量：
DFS 时记录 dfn_u 表示到达节点 u 时的时间，而 low_u 表示 u 点能够访问到的所有点的最小 dfn。同时，我们维护一个栈表示目前访问过，且没有被分入强连通分量的点。

强连通分量

强连通分量是指有向图的一个子图，满足子图中所有点之间可以两两到达。我们可以用 Tarjan 算法来求出所有的极大强连通分量：
DFS 时记录 dfn_u 表示到达节点 u 时的时间，而 low_u 表示 u 点能够访问到的所有点的最小 dfn。同时，我们维护一个栈表示目前访问过，且没有被分入强连通分量的点。
显然有 $low_u \leq dfn_u$。若 $low_u = dfn_u$，则说明 DFS 树中，节点 u 的子树中没有被分入强连通分量的节点应该组成一个强连通分量。证明是 trival 的。

强连通分量

强连通分量是指有向图的一个子图，满足子图中所有点之间可以两两到达。我们可以用 Tarjan 算法来求出所有的极大强连通分量：
DFS 时记录 dfn_u 表示到达节点 u 时的时间，而 low_u 表示 u 点能够访问到的所有点的最小 dfn。同时，我们维护一个栈表示目前访问过，且没有被分入强连通分量的点。
显然有 $low_u \le dfn_u$。若 $low_u = dfn_u$，则说明 DFS 树中，节点 u 的子树中没有被分入强连通分量的节点应该组成一个强连通分量。证明是 trival 的。
因此，当 $low_u = dfn_u$ 时，我们应当不断弹栈，直到 u 被弹出。弹出的所有节点就组成了一个强连通分量。

双连通分量

双连通分量是对无向图定义的，有点双连通分量和边双连通分量两种。

双连通分量

双连通分量是对无向图定义的，有点双连通分量和边双连通分量两种。

点双连通分量指一个子图，满足任意删去一个点后这个子图仍连通。

双连通分量

双连通分量是对无向图定义的，有点双连通分量和边双连通分量两种。

点双连通分量指一个子图，满足任意删去一个点后这个子图仍连通。
我们仍然可以使用 Tarjan 算法求出所有极大的点双连通分量：

双连通分量

双连通分量是对无向图定义的，有点双连通分量和边双连通分量两种。

点双连通分量指一个子图，满足任意删去一个点后这个子图仍连通。
我们仍然可以使用 Tarjan 算法求出所有极大的点双连通分量：
我们仍然记录 dfn_u 和 low_u，含义同强连通分量的 Tarjan 算法。

双连通分量

双连通分量是对无向图定义的，有点双连通分量和边双连通分量两种。

点双连通分量指一个子图，满足任意删去一个点后这个子图仍连通。
我们仍然可以使用 Tarjan 算法求出所有极大的点双连通分量：
我们仍然记录 dfn_u 和 low_u，含义同强连通分量的 Tarjan 算法。
如果某个点 u 的孩子 v 满足 $low_v \geq dfn_u$，那么我们就不断弹栈，直到栈顶元素为 u。则此时弹出的所有元素组成了一个点双连通分量。

双连通分量

双连通分量是对无向图定义的，有点双连通分量和边双连通分量两种。

点双连通分量指一个子图，满足任意删去一个点后这个子图仍连通。
我们仍然可以使用 Tarjan 算法求出所有极大的点双连通分量：
我们仍然记录 dfn_u 和 low_u，含义同强连通分量的 Tarjan 算法。
如果某个点 u 的孩子 v 满足 $low_v \geq dfn_u$，那么我们就不断弹栈，直到栈顶元素为 u。则此时弹出的所有元素组成了一个点双连通分量。
值得一提的是，这里 u 并不弹出，因为**一个点可能同时位于**多个点双连通分量中。

双连通分量

双连通分量是对无向图定义的，有点双连通分量和边双连通分量两种。

点双连通分量指一个子图，满足任意删去一个点后这个子图仍连通。
我们仍然可以使用 Tarjan 算法求出所有极大的点双连通分量：
我们仍然记录 dfn_u 和 low_u，含义同强连通分量的 Tarjan 算法。
如果某个点 u 的孩子 v 满足 $low_v \geq dfn_u$，那么我们就不断弹栈，直到栈顶元素为 u。则此时弹出的所有元素组成了一个点双连通分量。
值得一提的是，这里 u 并不弹出，因为**一个点可能同时位于**多个点双连通分量中。

边双连通分量指一个子图，满足任意删去一条边后这个子图仍连通。

双连通分量

双连通分量是对无向图定义的，有点双连通分量和边双连通分量两种。

点双连通分量指一个子图，满足任意删去一个点后这个子图仍连通。
我们仍然可以使用 Tarjan 算法求出所有极大的点双连通分量：
我们仍然记录 dfn_u 和 low_u，含义同强连通分量的 Tarjan 算法。
如果某个点 u 的孩子 v 满足 $low_v \geq dfn_u$，那么我们就不断弹栈，直到栈顶元素为 u。则此时弹出的所有元素组成了一个点双连通分量。
值得一提的是，这里 u 并不弹出，因为**一个点可能同时位于**多个点双连通分量中。

边双连通分量指一个子图，满足任意删去一条边后这个子图仍连通。
边双连通分量的 Tarjan 算法和强连通分量类似，因为将树边看成向下的，返祖边看成向上的，即可把求边双连通分量看做求强连通分量的问题了。

World Finals 2011 H Mining Your Own Business

给定 n 个点 m 条边的无向图，你要标关键点，使得删去任意一个点后其他所有点都能到达至少一个关键点。
最小化关键点个数，并求出最小的方案数。
复杂度要求线性。

UOJ Goodbye Jiawu B

给定 n 个点 m 条边的无向图，求哪些点删掉之后这个图会变成一棵树。复杂度要求线性。

BJOI 2013 压力

给定 n 个点 m 条边的无向图，每次操作将 u, v 之间所有必须经过的点权值加一，求操作完了之后每个点的权值。
$n \le 10^5$，$m, q \le 2 \times 10^6$。

定义

有 n 个非负整数变量和 m 个不等式，每个不等式形如 $x_i - x_j \geq k$。求出是否存在可行解，如果存在，求出 $\sum_{i=1}^{n} x_i$ 的最小值。

求解

为每个 x_i 建立一个点，对于 $x_i \geq x_j + k$，在图中连接 $x_j \xrightarrow{k} x_i$ 的边。

求解

为每个 x_i 建立一个点，对于 $x_i \geq x_j + k$，在图中连接 $x_j \xrightarrow{k} x_i$ 的边。显然，如果该图中存在正环，那么无解。否则，建立超级源点 S 并向每个点连接权值为 0 的边，则每个点的最小值是 S 到它的最长路。且容易归纳证明该下界可以取到。

求解

为每个 x_i 建立一个点，对于 $x_i \geq x_j + k$，在图中连接 $x_j \xrightarrow{k} x_i$ 的边。显然，如果该图中存在正环，那么无解。否则，建立超级源点 S 并向每个点连接权值为 0 的边，则每个点的最小值是 S 到它的最长路。且容易归纳证明该下界可以取到。
最长路可以边权取反后通过 Bellman-Ford 算法求出。

POI 2012 Festival

n 个变量 m 个条件，每个形如 $x_i = x_j + 1$ 或 $x_i \leq x_j$，求所有变量不同取值个数的最大值。
$n \leq 600$。

SCOI 2011 糖果

n 个变量 m 个条件，每个形如 $x_i = x_j$， $x_i \le x_j$， $x_i \ge x_j$， $x_i < x_j$， $x_i > x_j$， 求和最小的非负整数解。
$n, m \le 10^5$。

定义

有 n 个 $0/1$ 变量和 m 个等式，每个等式形如 $x_i \operatorname{op} x_j = k$，其中 op 是一个位运算。
你要求出一组可行解。

求解

将每个 x_i 拆成两个点 x_{i0} 以及 x_{i1}，分别表示 x_i 取 0 或者 1。

求解

将每个 x_i 拆成两个点 x_{i0} 以及 x_{i1}，分别表示 x_i 取 0 或者 1。
每个限制条件可以被转化为若干新的限制，形如：如果 x_a 是 p，那么 x_b 必须要是 q，其中 $a, b \in \{i, j\}$，而 $p, q \in \{0, 1\}$。对于每个这样的限制，我们将 x_{ap} 向 x_{bq} 连一条边，表示选了 x_{ap} 就必须也要选 x_{bq}。

求解

将每个 x_i 拆成两个点 x_{i0} 以及 x_{i1}，分别表示 x_i 取 0 或者 1。

每个限制条件可以被转化为若干新的限制，形如：如果 x_a 是 p，那么 x_b 必须要是 q，其中 $a, b \in \{i, j\}$，而 $p, q \in \{0, 1\}$。对于每个这样的限制，我们将 x_{ap} 向 x_{bq} 连一条边，表示选了 x_{ap} 就必须也要选 x_{bq}。

考虑新图的每个强连通分量，显然每个强连通分量要么一起选择，要么一起不选。因此，在缩点后我们可以得到一个新图，这个新图是有向无环图。

求解

将每个 x_i 拆成两个点 x_{i0} 以及 x_{i1}，分别表示 x_i 取 0 或者 1。
每个限制条件可以被转化为若干新的限制，形如：如果 x_a 是 p，那么 x_b 必须要是 q，其中 $a, b \in \{i, j\}$，而 $p, q \in \{0, 1\}$。对于每个这样的限制，我们将 x_{ap} 向 x_{bq} 连一条边，表示选了 x_{ap} 就必须也要选 x_{bq}。
考虑新图的每个强连通分量，显然每个强连通分量要么一起选择，要么一起不选。因此，在缩点后我们可以得到一个新图，这个新图是有向无环图。显然如果 x_{i0} 和 x_{i1} 在同一个强连通分量里面，那么一定无解，因为这两个点应该选恰好一个。

求解

将每个 x_i 拆成两个点 x_{i0} 以及 x_{i1}，分别表示 x_i 取 0 或者 1。
每个限制条件可以被转化为若干新的限制，形如：如果 x_a 是 p，那么 x_b 必须要是 q，其中 $a, b \in \{i, j\}$，而 $p, q \in \{0, 1\}$。对于每个这样的限制，我们将 x_{ap} 向 x_{bq} 连一条边，表示选了 x_{ap} 就必须也要选 x_{bq}。
考虑新图的每个强连通分量，显然每个强连通分量要么一起选择，要么一起不选。因此，在缩点后我们可以得到一个新图，这个新图是有向无环图。显然如果 x_{i0} 和 x_{i1} 在同一个强连通分量里面，那么一定无解，因为这两个点应该选恰好一个。考虑如何构造方案：
考虑贪心，按照逆拓扑序依次考虑每个点，如果这个点不能选，那么我们跳过他。否则，我们选择这个点，然后把这个强连通分量里面所有点对应的另一个点标记为不可以选。

求解

将每个 x_i 拆成两个点 x_{i0} 以及 x_{i1}，分别表示 x_i 取 0 或者 1。
每个限制条件可以被转化为若干新的限制，形如：如果 x_a 是 p，那么 x_b 必须要是 q，其中 $a,b \in \{i,j\}$，而 $p,q \in \{0,1\}$。对于每个这样的限制，我们将 x_{ap} 向 x_{bq} 连一条边，表示选了 x_{ap} 就必须也要选 x_{bq}。
考虑新图的每个强连通分量，显然每个强连通分量要么一起选择，要么一起不选。因此，在缩点后我们可以得到一个新图，这个新图是有向无环图。显然如果 x_{i0} 和 x_{i1} 在同一个强连通分量里面，那么一定无解，因为这两个点应该选恰好一个。考虑如何构造方案：
考虑贪心，按照逆拓扑序依次考虑每个点，如果这个点不能选，那么我们跳过他。否则，我们选择这个点，然后把这个强连通分量里面所有点对应的另一个点标记为不可以选。
因为 2-SAT 问题具有对称性：即如果有边 $x_{ap} \to y_{bq}$，那么一定有 $y_{b\neg q} \to x_{a\neg p}$ 的边，所以这样构造方案是对的。

JSOI 2010 满汉全席

n 道菜，每道菜可以做成汉族口味和满族口味。m 个评委，每个评委会对两道不同的菜有要求。要求都是某个菜要做成什么口味。问是否存在一种方案，使得每个评委至少有一个要求被满足。

$n \le 100$，$m \le 1000$，$T \le 50$。

NOI 2017 游戏

有 n 场游戏和三种车，每个游戏可以选择用一种车，每个游戏可能要求不能使用某种车，也可能没有要求。
给出 m 个要求，表示如果第 i 个游戏用了 x 车，那么第 j 个游戏要用 y 车。求一种合法方案。
$n \leq 5 \times 10^4$，$m \leq 10^5$，没有要求的游戏个数不超过 8。

定义

独立集是指点集 V 的一个子集 S，满足 $\forall u, v \in S, (u, v) \notin E$。

定义

独立集是指点集 V 的一个子集 S，满足 $\forall u, v \in S, (u, v) \notin E$。
团是指点集 V 的一个子集 S，满足 $\forall u, v \in S, (u, v) \in E$。

定义

独立集是指点集 V 的一个子集 S，满足 $\forall u, v \in S, (u, v) \notin E$。
团是指点集 V 的一个子集 S，满足 $\forall u, v \in S, (u, v) \in E$。
最大独立集和最大团问题都是 NP Complete 问题。

定义

独立集是指点集 V 的一个子集 S，满足 $\forall u, v \in S, (u, v) \notin E$。
团是指点集 V 的一个子集 S，满足 $\forall u, v \in S, (u, v) \in E$。
最大独立集和最大团问题都是 NP Complete 问题。~~如果有人会做就可以去拿图灵奖了。~~

定义

独立集是指点集 V 的一个子集 S，满足 $\forall u, v \in S, (u, v) \notin E$。
团是指点集 V 的一个子集 S，满足 $\forall u, v \in S, (u, v) \in E$。
最大独立集和最大团问题都是 NP Complete 问题。~~如果有人会做就可以去拿图灵奖了。~~
点覆盖是指点集 V 的一个子集 S，满足 $\forall (u, v) \in E, u \in S \vee v \in S$。

定义

独立集是指点集 V 的一个子集 S，满足 $\forall u, v \in S, (u, v) \notin E$。
团是指点集 V 的一个子集 S，满足 $\forall u, v \in S, (u, v) \in E$。
最大独立集和最大团问题都是 NP Complete 问题。~~如果有人会做就可以去拿图灵奖了。~~
点覆盖是指点集 V 的一个子集 S，满足 $\forall (u, v) \in E, u \in S \vee v \in S$。
支配集是指点集 V 的一个子集 S，满足
$\forall u \in V \setminus S, \exists v \in S, \text{s.t.} (u, v) \in E$。

定义

独立集是指点集 V 的一个子集 S，满足 $\forall u, v \in S, (u, v) \notin E$。
团是指点集 V 的一个子集 S，满足 $\forall u, v \in S, (u, v) \in E$。
最大独立集和最大团问题都是 NP Complete 问题。~~如果有人会做就可以去拿图灵奖了。~~
点覆盖是指点集 V 的一个子集 S，满足 $\forall (u, v) \in E, u \in S \vee v \in S$。
支配集是指点集 V 的一个子集 S，满足 $\forall u \in V \setminus S, \exists v \in S, \text{s.t.} (u, v) \in E$。
因为最小点覆盖与最大独立集之和就是点数，而最小支配集问题也可以规约到最小点覆盖问题，所以最小点覆盖问题和最小支配集问题也都是 NP Complete 问题。

定义

一笔画问题：判断一个图是否是一个能够遍历完所有的边而没有重复的图。

定义

一笔画问题：判断一个图是否是一个能够遍历完所有的边而没有重复的图。
这样的图称为**欧拉图**，这时遍历的路径称作**欧拉路径**。如果路径是一个环，则称为**欧拉回路**。

判断

联通无向图有**欧拉路径**的充要条件是：图中奇顶点的数目等于 0 或者 2；存在**欧拉回路**的充要条件是：每个顶点的度都是偶数。

判断

联通无向图有**欧拉路径**的充要条件是：图中奇顶点的数目等于 0 或者 2；存在**欧拉回路**的充要条件是：每个顶点的度都是偶数。

证明.

必要性：如果一个图能一笔画成，那么对每一个顶点，要么路径中“进入”这个点的边数等于“离开”这个点的边数：这时点的度为偶数。要么两者相差一：这时这个点必然是起点或终点之一。注意到有起点就必然有终点，因此奇顶点的数目要么是 0，要么是 2。

判断

联通无向图有**欧拉路径**的充要条件是：图中奇顶点的数目等于 0 或者 2；存在**欧拉回路**的充要条件是：每个顶点的度都是偶数。

证明.

必要性： 如果一个图能一笔画成，那么对每一个顶点，要么路径中“进入”这个点的边数等于“离开”这个点的边数：这时点的度为偶数。要么两者相差一：这时这个点必然是起点或终点之一。注意到有起点就必然有终点，因此奇顶点的数目要么是 0，要么是 2。
充分性： 如果图中没有奇顶点，那么随便选一个点出发，连一个环 C_1。如果这个环就是原图，则结束这一过程。如果不是，由于原图是连通的，C_1 和原图的其它部分必然有公共顶点 s_1。从这一点出发，在原图的剩余部分中重复上述步骤。由于原图是连通图，经过若干步后，全图被分为一些环。因为两个相连的环就是一个环，所以原图为欧拉回路。

判断

联通无向图有**欧拉路径**的充要条件是：图中奇顶点的数目等于 0 或者 2；存在**欧拉回路**的充要条件是：每个顶点的度都是偶数。

证明.

必要性： 如果一个图能一笔画成，那么对每一个顶点，要么路径中"进入"这个点的边数等于"离开"这个点的边数：这时点的度为偶数。要么两者相差一：这时这个点必然是起点或终点之一。注意到有起点就必然有终点，因此奇顶点的数目要么是 0，要么是 2。

充分性： 如果图中没有奇顶点，那么随便选一个点出发，连一个环 C_1。如果这个环就是原图，则结束这一过程。如果不是，由于原图是连通的，C_1 和原图的其它部分必然有公共顶点 s_1。从这一点出发，在原图的剩余部分中重复上述步骤。由于原图是连通图，经过若干步后，全图被分为一些环。因为两个相连的环就是一个环，所以原图为欧拉回路。如果图中有两个奇顶点 u 和 v，那么加多一条边将它们连上后得到一个无奇顶点的连通图。由上知这个图是一个环，因此去掉新加的边后成为一条路径，起点和终点是 u 和 v。 $\square$

求解

考虑充分性的证明过程，可以得到如下解法：从任意非孤立点开始 DFS，每次我们从当前节点的边中任选一条，将其删除，然后进入这条边的另一端继续进行 DFS。则最终所有边一定会被经过，且在回溯时记录经过的边的编号，其**逆序**即为欧拉回路。

求解

考虑充分性的证明过程，可以得到如下解法：从任意非孤立点开始 DFS，每次我们从当前节点的边中任选一条，将其删除，然后进入这条边的另一端继续进行 DFS。则最终所有边一定会被经过，且在回溯时记录经过的边的编号，其**逆序**即为欧拉回路。伪代码如下：

算法 求无向图的欧拉回路

function $\textsc{Solve}(u)$ ▷ 从节点 u 开始 DFS
  while E 中存在至少一条以 u 为端点的边 **do**
    任取 $e = (u, v) \in E$ ▷ 任取一条以 u 为端点的边
    从 E 中删除 e ▷ 删去这条边
    $\textsc{Solve}(v)$ ▷ 对边的另一端点进行递归
    $\textsc{PushFront}(P, e)$ ▷ 将边 e 加入答案头部
  end while
end function
$\textsc{Solve}(u)$ ▷ 这里的 $u \in V$ 为任意非孤立点

POI 2011 Party

给定一张 n 个点 m 条边的无向图，保证图中存在大小至少为 ${}^{2n}/_{3}$ 的团，求一个大小为 ${}^{n}/_{3}$ 的团。
$n \le 3000$，$3 \mid n$。

POI 2011 Conspriracy

给定一张 n 个点 m 条边的无向图，你要把 V 分为 S 和 $T = V \setminus S$ 两部分，使得 $S, T \neq \varnothing$，且 S 是团而 T 是独立集。求方案数。

$n \leq 5000$。

POI 2011 Garbage

给定 n 个点 m 条边的无向图，初始时每条边的权值都是 0 或 1，每次可以选择一个简单环，将环上的所有边权值异或 1。
给定最后每条边的权值，构造一个操作方案。
复杂度要求线性。

Thank You!