图论模型的建立

dy0607

雅礼中学

January 14, 2019

Preface

由于只有一个下午的时间，这里只会涉及部分常见(?)的图论模型。

Preface

由于只有一个下午的时间，这里只会涉及部分常见(?)的图论模型。

有其他一些算法虽然也可能会用到，但由于（太水/太偏/讲题人忘了）所以这里不会出现。

Preface

由于只有一个下午的时间，这里只会涉及部分常见(?)的图论模型。

有其他一些算法虽然也可能会用到，但由于（太水/太偏/讲题人忘了）所以这里不会出现。

题目都很可做，欢迎大家上来切题。

Overview

Matchings

- 匹配主要是二分图匹配，设最大匹配为P，与此相关的几个结论：

Matchings

- 匹配主要是二分图匹配，设最大匹配为P，与此相关的几个结论：
  - 最小边覆盖 $= n - P$.

Matchings

- 匹配主要是二分图匹配，设最大匹配为P，与此相关的几个结论：
  - 最小边覆盖 $= n - P$.
  - 最大独立集 $= n - P$.

Matchings

- 匹配主要是二分图匹配，设最大匹配为P，与此相关的几个结论：
  - 最小边覆盖 $= n - P$.
  - 最大独立集 $= n - P$.
  - 最小点覆盖 $= P$.

Matchings

- 匹配主要是二分图匹配，设最大匹配为P，与此相关的几个结论：
  - 最小边覆盖 $= n - P$.
  - 最大独立集 $= n - P$.
  - 最小点覆盖 $= P$.
- Hall定理：二分图$G = (X, Y, E)$中，设$f(S)$为与点集S中的点有连边的Y集合点的数量。那么G有完美匹配当且仅当:

Matchings

- 匹配主要是二分图匹配，设最大匹配为P，与此相关的几个结论：
  - 最小边覆盖 $= n - P$.
  - 最大独立集 $= n - P$.
  - 最小点覆盖 $= P$.
- Hall定理：二分图$G = (X, Y, E)$中，设$f(S)$为与点集S中的点有连边的Y集合点的数量。那么G有完美匹配当且仅当:

$$\forall S \in X, f(S) \geq |S|$$

Matchings

- 匹配主要是二分图匹配，设最大匹配为P，与此相关的几个结论：
  - 最小边覆盖 $= n - P$.
  - 最大独立集 $= n - P$.
  - 最小点覆盖 $= P$.
- Hall定理：二分图$G = (X, Y, E)$中，设$f(S)$为与点集S中的点有连边的Y集合点的数量。那么G有完美匹配当且仅当:

$$\forall S \in X, f(S) \geq |S|$$

必要性显然，充分性可以用反证法：如果不存在完美匹配，则可以通过这个条件找出一条新的增广路。

Directed Acyclic Graph

- 最小路径覆盖：用最少的路径覆盖所有点恰好一次。

Directed Acyclic Graph

- 最小路径覆盖：用最少的路径覆盖所有点恰好一次。
  - 将每个点u拆成两个点u_1, u_2，然后对于边(u, v)，连边(u_1, v_2)。

Directed Acyclic Graph

- 最小路径覆盖：用最少的路径覆盖所有点恰好一次。
  - 将每个点u拆成两个点u_1, u_2，然后对于边(u, v)，连边(u_1, v_2)。
  - 求最大匹配P，最小路径覆盖就是$n - P$。这里两个点匹配即代表它们在一条路径上。

Directed Acyclic Graph

- 最小路径覆盖：用最少的路径覆盖所有点恰好一次。
  - 将每个点u拆成两个点u_1, u_2，然后对于边(u, v)，连边(u_1, v_2)。
  - 求最大匹配P，最小路径覆盖就是$n - P$。这里两个点匹配即代表它们在一条路径上。
- 最小链覆盖：用最少的链覆盖所有点至少一次。

Directed Acyclic Graph

- 最小路径覆盖：用最少的路径覆盖所有点恰好一次。
  - 将每个点u拆成两个点u_1, u_2，然后对于边(u, v)，连边(u_1, v_2)。
  - 求最大匹配P，最小路径覆盖就是$n - P$。这里两个点匹配即代表它们在一条路径上。
- 最小链覆盖：用最少的链覆盖所有点至少一次。
  - 做传递闭包后可转化为最小路径覆盖。（也可以直接网络流）

Directed Acyclic Graph

- 最小路径覆盖：用最少的路径覆盖所有点恰好一次。
  - 将每个点u拆成两个点u_1, u_2，然后对于边(u, v)，连边(u_1, v_2)。
  - 求最大匹配P，最小路径覆盖就是$n - P$。这里两个点匹配即代表它们在一条路径上。
- 最小链覆盖：用最少的链覆盖所有点至少一次。
  - 做传递闭包后可转化为最小路径覆盖。（也可以直接网络流）
- Dilworth定理：有向无环图中，最长反链等于最小链覆盖。如何构造/证明?

Directed Acyclic Graph

- 最小路径覆盖：用最少的路径覆盖所有点恰好一次。
  - 将每个点u拆成两个点u_1, u_2，然后对于边(u, v)，连边(u_1, v_2)。
  - 求最大匹配P，最小路径覆盖就是$n - P$。这里两个点匹配即代表它们在一条路径上。
- 最小链覆盖：用最少的链覆盖所有点至少一次。
  - 做传递闭包后可转化为最小路径覆盖。（也可以直接网络流）
- Dilworth定理：有向无环图中，最长反链等于最小链覆盖。如何构造/证明?
  - 求出最小链覆盖建出的二分图G中的最大独立集，对于一个点u，如果u_1, u_2都在独立集内，则将u加入反链。

Directed Acyclic Graph

- 最小路径覆盖：用最少的路径覆盖所有点恰好一次。
  - 将每个点u拆成两个点u_1, u_2，然后对于边(u, v)，连边(u_1, v_2)。
  - 求最大匹配P，最小路径覆盖就是$n - P$。这里两个点匹配即代表它们在一条路径上。
- 最小链覆盖：用最少的链覆盖所有点至少一次。
  - 做传递闭包后可转化为最小路径覆盖。（也可以直接网络流）
- Dilworth定理：有向无环图中，最长反链等于最小链覆盖。如何构造/证明?
  - 求出最小链覆盖建出的二分图G中的最大独立集，对于一个点u，如果u_1, u_2都在独立集内，则将u加入反链。
  - 设G的最大匹配为P，那么独立集大小为$2n - P$，满足条件的u至少有$n - P$个。也就是求出的反链大于等于最小链覆盖，而显然有最长反链小于等于最小链覆盖，定理得证。

Codeforces 1054F

平面上有若干条线段，有红蓝两种颜色。红色线段与X轴平行，蓝色线段与Y轴平行，同色线段均不相交。

已知这些线段之间形成了n个交点（在端点相交也算相交），给出这些交点，求平面上至少有多少条线段，并输出任意一种方案。线段长度允许为0。

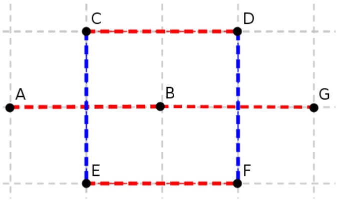
$n \le 10^3, x_i, y_i \le 10^9$

Solution

初始时在每个交点放两条长度为0的异色线段，构造出所有交点。
为了使线段尽量少，我们可以用一条线段覆盖多个点：

Solution

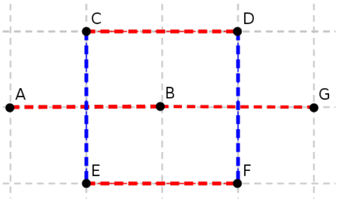
初始时在每个交点放两条长度为0的异色线段，构造出所有交点。为了使线段尽量少，我们可以用一条线段覆盖多个点：



不难发现问题就是保留尽量多的线段，使得它们不在除端点以外的位置相交。

Solution

初始时在每个交点放两条长度为0的异色线段，构造出所有交点。为了使线段尽量少，我们可以用一条线段覆盖多个点：

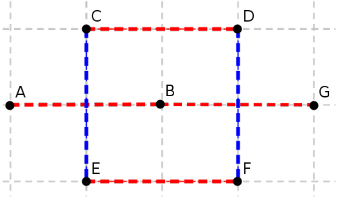


不难发现问题就是保留尽量多的线段，使得它们不在除端点以外的位置相交。

将线段视为点，将冲突视为边，只需求这个二分图的最大独立集。

Solution

初始时在每个交点放两条长度为0的异色线段，构造出所有交点。
为了使线段尽量少，我们可以用一条线段覆盖多个点：



不难发现问题就是保留尽量多的线段，使得它们不在除端点以外的位置相交。

将线段视为点，将冲突视为边，只需求这个二分图的最大独立集。

$O(n^3)$，大约有一个$\frac{1}{16}$的常数。

Codeforces 1009G

有一个长为n的字符串S，每个字符在a到f之间。你知道每个字符在S中的出现次数，以及S的每个位置可能出现哪些字符。求字典序最小的满足条件的字符串S，或判断无解。

$n \leq 10^5$

Solution

从前往后贪心，每次确定一个位置的字符，贪心时需要确保确定完一个位置之后后缀有解。

Solution

从前往后贪心，每次确定一个位置的字符，贪心时需要确保确定完一个位置之后后缀有解。

不难发现这是一个位置与字符之间的二分图匹配，判断是否有完美匹配可以用Hall定理来解决。

Solution

从前往后贪心，每次确定一个位置的字符，贪心时需要确保确定完一个位置之后后缀有解。

不难发现这是一个位置与字符之间的二分图匹配，判断是否有完美匹配可以用Hall定理来解决。

对于每个字符的集合s，预处理后缀中能填s中任意一个字符的位置有多少个，只需保证可用位置个数不少于s的字符可用个数之和。

Solution

从前往后贪心，每次确定一个位置的字符，贪心时需要确保确定完一个位置之后后缀有解。

不难发现这是一个位置与字符之间的二分图匹配，判断是否有完美匹配可以用Hall定理来解决。

对于每个字符的集合s，预处理后缀中能填s中任意一个字符的位置有多少个，只需保证可用位置个数不少于s的字符可用个数之和。

$O(n2^k)$，其中k是字符集大小。

Codeforces 590E

给出n个仅含a/b的串，从中选出尽量多的串，使得串之间不存在包含关系。

$n \leq 750, \sum |S| \leq 10^7$

Solution

先把重复的串去掉。用AC自动机可以求出，对于一个串S_i的每个前缀，是否有一个其他的串S_j是这个前缀的后缀，以及这样的S_j中长度最大的一个S_k。

Solution

先把重复的串去掉。用AC自动机可以求出，对于一个串S_i的每个前缀，是否有一个其他的串S_j是这个前缀的后缀，以及这样的S_j中长度最大的一个S_k。

连边(i,k)，做传递闭包后就可以知道每个串之间的包含关系。

Solution

先把重复的串去掉。用AC自动机可以求出，对于一个串S_i的每个前缀，是否有一个其他的串S_j是这个前缀的后缀，以及这样的S_j中长度最大的一个S_k。

连边(i,k)，做传递闭包后就可以知道每个串之间的包含关系。

然后就是一个最长反链问题，$O(n^3+\sum|S|)$。

Flows

- Dinic的复杂度为$O(n^2m)$，但通常复杂度要优秀很多，例如Dinic跑二分匹配的复杂度为$O(\sqrt{n}m)$。

Flows

- Dinic的复杂度为$O(n^2m)$，但通常复杂度要优秀很多，例如Dinic跑二分匹配的复杂度为$O(\sqrt{n}m)$。
  - Dinic算法中，每次会找到长度最短的增广路，并将这个长度的增广路全部增广完。因此下一次增广路长度至少$+1$。

Flows

- Dinic的复杂度为$O(n^2m)$，但通常复杂度要优秀很多，例如Dinic跑二分匹配的复杂度为$O(\sqrt{n}m)$。
  - Dinic算法中，每次会找到长度最短的增广路，并将这个长度的增广路全部增广完。因此下一次增广路长度至少$+1$。
  - 前$\sqrt{n}$次增广后增广路长度至少为$\sqrt{n}$，此时找到的匹配和最优解之间的差异可以看成若干条增广路，这些路径不相交且长度大于$\sqrt{n}$，因此其数量不超过$\sqrt{n}$。那么再进行$\sqrt{n}$次增广一定能找到最优解。

Flows

- Dinic的复杂度为$O(n^2m)$，但通常复杂度要优秀很多，例如Dinic跑二分匹配的复杂度为$O(\sqrt{n}m)$。
  - Dinic算法中，每次会找到长度最短的增广路，并将这个长度的增广路全部增广完。因此下一次增广路长度至少+1。
  - 前$\sqrt{n}$次增广后增广路长度至少为$\sqrt{n}$，此时找到的匹配和最优解之间的差异可以看成若干条增广路，这些路径不相交且长度大于$\sqrt{n}$，因此其数量不超过$\sqrt{n}$。那么再进行$\sqrt{n}$次增广一定能找到最优解。
- 最大权闭合子图：对于带点权有向图G，找出一个权值和最大的子图，满足所有子图中的点，其指向的节点也在子图中。

Flows

- Dinic的复杂度为$O(n^2m)$，但通常复杂度要优秀很多，例如Dinic跑二分匹配的复杂度为$O(\sqrt{n}m)$。
    - Dinic算法中，每次会找到长度最短的增广路，并将这个长度的增广路全部增广完。因此下一次增广路长度至少$+1$。
    - 前$\sqrt{n}$次增广后增广路长度至少为$\sqrt{n}$，此时找到的匹配和最优解之间的差异可以看成若干条增广路，这些路径不相交且长度大于$\sqrt{n}$，因此其数量不超过$\sqrt{n}$。那么再进行$\sqrt{n}$次增广一定能找到最优解。
- 最大权闭合子图：对于带点权有向图G，找出一个权值和最大的子图，满足所有子图中的点，其指向的节点也在子图中。
    - 从S向正权点连容量为权值的边，负权点向T连容量为权值相反数的边，其余的边容量为∞。做最小割即可。

Flows

- Dinic的复杂度为$O(n^2m)$，但通常复杂度要优秀很多，例如Dinic跑二分匹配的复杂度为$O(\sqrt{n}m)$。
  - Dinic算法中，每次会找到长度最短的增广路，并将这个长度的增广路全部增广完。因此下一次增广路长度至少+1。
  - 前$\sqrt{n}$次增广后增广路长度至少为$\sqrt{n}$，此时找到的匹配和最优解之间的差异可以看成若干条增广路，这些路径不相交且长度大于$\sqrt{n}$，因此其数量不超过$\sqrt{n}$。那么再进行$\sqrt{n}$次增广一定能找到最优解。
- 最大权闭合子图：对于带点权有向图G，找出一个权值和最大的子图，满足所有子图中的点，其指向的节点也在子图中。
  - 从S向正权点连容量为权值的边，负权点向T连容量为权值相反数的边，其余的边容量为∞。做最小割即可。
- 平面图最小割可以转化为对偶图上的最短路问题。

Flows with lower bounds

- 无源汇可行流：建立超级源点汇点S, T，对于一条下界为l上界为r的边(u, v)，则连边$(u, T, l), (u, v, r-l), (S, v, l)$。然后从$S$到$T$做最大流，满流则合法。

Flows with lower bounds

- 无源汇可行流：建立超级源点汇点S,T，对于一条下界为l上界为r的边(u,v)，则连边$(u,T,l),(u,v,r-l),(S,v,l)$。然后从$S$到$T$做最大流，满流则合法。
- 有源汇可行流：从原来的汇点t向源点s连一条容量∞的边，转化为无源汇可行流。

Flows with lower bounds

- 无源汇可行流：建立超级源点汇点S,T，对于一条下界为l上界为r的边(u,v)，则连边$(u,T,l),(u,v,r-l),(S,v,l)$。然后从$S$到$T$做最大流，满流则合法。
- 有源汇可行流：从原来的汇点t向源点s连一条容量∞的边，转化为无源汇可行流。
- 最大可行流：先做出可行流，然后将t到s连的边删掉，最后从s到t做最大流。

Flows with lower bounds

- 无源汇可行流：建立超级源点汇点S, T，对于一条下界为l上界为r的边(u, v)，则连边$(u, T, l), (u, v, r-l), (S, v, l)$。然后从$S$到$T$做最大流，满流则合法。
- 有源汇可行流：从原来的汇点t向源点s连一条容量∞的边，转化为无源汇可行流。
- 最大可行流：先做出可行流，然后将t到s连的边删掉，最后从s到t做最大流。

2018集训队作业 Roundtrip

一个n个点m条边的无向图，现在希望从a号点开始，按顺序经过b号点和c号点，最后回到a。找出一条这样的路径，经过每个点最多一次(这里从a号点开始不算经过a号点)。或者判断无解。

$n, m \leq 10^5$，给出的图保证存在3条从a到c的不相交简单路径。

Solution

考虑用网络流先把三条路径找到。将每个点拆成两个点，一个点连入边，另一个点连出边，中间流量为1，从a到c做网络流，这里并不需要做最大流，只要流量为3即可。

Solution

考虑用网络流先把三条路径找到。将每个点拆成两个点，一个点连入边，另一个点连出边，中间流量为1，从a到c做网络流，这里并不需要做最大流，只要流量为3即可。

如果b在这三条路径上就做完了，否则考虑找两条从b到这三条路径上的点的不相交简单路径，可以发现如果有解则一定能找到一条路径，需要讨论一些情况。

Solution

考虑用网络流先把三条路径找到。将每个点拆成两个点，一个点连入边，另一个点连出边，中间流量为1，从a到c做网络流，这里并不需要做最大流，只要流量为3即可。

如果b在这三条路径上就做完了，否则考虑找两条从b到这三条路径上的点的不相交简单路径，可以发现如果有解则一定能找到一条路径，需要讨论一些情况。

$O(n+m)$

LOJ 6079

你养了一只猫，为了让它快乐地成长，你需要合理地安排它每天的作息时间。假设一天分为 n 个时刻，猫在每个时刻要么是吃东西，要么是睡觉。在第 i 个时刻，假如猫是去吃东西，那么它能获得愉悦值 e_i，假如是去睡觉，那么能获得的愉悦值为 s_i。

经过研究，对于每一个连续的长度为 k 的作息区间，猫都要至少有 ms 的时刻用来睡觉，me 的时刻用来吃东西。最大化愉悦值并输出方案。

$n \leq 1000$，一个subtask是$ms = 0$。

Solution

设sum_i为$[\max(i-k+1,1),i]$中选择睡觉的次数，x_i表示i时刻是否选择吃饭，那么：

- $sum_i = sum_{i-1} + 1 - x_i, i \in [1,k]$
- $sum_i = sum_{i-1} + x_{i-k} - x_i, i \in [k+1,n]$

Solution

设sum_i为$[\max(i-k+1,1),i]$中选择睡觉的次数，x_i表示i时刻是否选择吃饭，那么：

- $sum_i = sum_{i-1} + 1 - x_i, i \in [1,k]$
- $sum_i = sum_{i-1} + x_{i-k} - x_i, i \in [k+1,n]$

- $sum_i + x_i - sum_{i-1} - 1 = 0, i \in [1,k]$
- $sum_i - sum_{i-1} + x_i - x_{i-k} = 0, i \in [k+1,n]$

Solution

设sum_i为$[\max(i-k+1,1),i]$中选择睡觉的次数，x_i表示i时刻是否选择吃饭，那么：

- $sum_i = sum_{i-1} + 1 - x_i, i \in [1,k]$
- $sum_i = sum_{i-1} + x_{i-k} - x_i, i \in [k+1,n]$

- $sum_i + x_i - sum_{i-1} - 1 = 0, i \in [1,k]$
- $sum_i - sum_{i-1} + x_i - x_{i-k} = 0, i \in [k+1,n]$

需要最大化$\sum_{i=1}^{n} s_i + x_i(e_i - s_i)$，有限制$ms \le sum_i \le k - me, k \le i \le n$.

Solution

设sum_i为$[\max(i-k+1,1),i]$中选择睡觉的次数，x_i表示i时刻是否选择吃饭，那么：

- $sum_i = sum_{i-1} + 1 - x_i, i \in [1,k]$
- $sum_i = sum_{i-1} + x_{i-k} - x_i, i \in [k+1,n]$

- $sum_i + x_i - sum_{i-1} - 1 = 0, i \in [1,k]$
- $sum_i - sum_{i-1} + x_i - x_{i-k} = 0, i \in [k+1,n]$

需要最大化$\sum_{i=1}^{n} s_i + x_i(e_i - s_i)$，有限制$ms \le sum_i \le k - me, k \le i \le n$.

注意到每个变量至多在方程中出现两次，且出现了两次的变量在两个方程中符号相反。

Solution

设sum_i为$[\max(i-k+1,1),i]$中选择睡觉的次数，x_i表示i时刻是否选择吃饭，那么：

- $sum_i = sum_{i-1} + 1 - x_i, i \in [1,k]$
- $sum_i = sum_{i-1} + x_{i-k} - x_i, i \in [k+1,n]$

- $sum_i + x_i - sum_{i-1} - 1 = 0, i \in [1,k]$
- $sum_i - sum_{i-1} + x_i - x_{i-k} = 0, i \in [k+1,n]$

需要最大化$\sum_{i=1}^{n} s_i + x_i(e_i - s_i)$，有限制$ms \le sum_i \le k - me, k \le i \le n$.

注意到每个变量至多在方程中出现两次，且出现了两次的变量在两个方程中符号相反。

Solution

在网络流中，如果将每条边的流量看成变量，由于入边总流量等于出边总流量，每个点就代表了一个等式。那么我们将变量看成边，将等式看成点，反过来就可以建出网络流。

Solution

在网络流中，如果将每条边的流量看成变量，由于入边总流量等于出边总流量，每个点就代表了一个等式。那么我们将变量看成边，将等式看成点，反过来就可以建出网络流。

- 对于一个出现了两次的变量，从这个变量取正号的等式向取负号的等式连边。

Solution

在网络流中，如果将每条边的流量看成变量，由于入边总流量等于出边总流量，每个点就代表了一个等式。那么我们将变量看成边，将等式看成点，反过来就可以建出网络流。

- 对于一个出现了两次的变量，从这个变量取正号的等式向取负号的等式连边。
- 对于只出现了一次的变量或者是常量，如果它取正号则由等式连向汇点；否则由源点连向等式。

Solution

在网络流中，如果将每条边的流量看成变量，由于入边总流量等于出边总流量，每个点就代表了一个等式。那么我们将变量看成边，将等式看成点，反过来就可以建出网络流。

- 对于一个出现了两次的变量，从这个变量取正号的等式向取负号的等式连边。
- 对于只出现了一次的变量或者是常量，如果它取正号则由等式连向汇点；否则由源点连向等式。
- 对于常量的边，网络流中要保证满流。

Solution

在网络流中，如果将每条边的流量看成变量，由于入边总流量等于出边总流量，每个点就代表了一个等式。那么我们将变量看成边，将等式看成点，反过来就可以建出网络流。

- 对于一个出现了两次的变量，从这个变量取正号的等式向取负号的等式连边。
- 对于只出现了一次的变量或者是常量，如果它取正号则由等式连向汇点；否则由源点连向等式。
- 对于常量的边，网络流中要保证满流。
- 给x_i对应的边加上费用$s_i - e_i$。

Solution

网络流可以处理对变量的上界限制，但这里对sum_i还有下界限制L。

Solution

网络流可以处理对变量的上界限制，但这里对sum_i还有下界限制L。

一种方法是用上下界处理，但可能出现负环，解决方法是初始做贪心来避免负权边。

Solution

网络流可以处理对变量的上界限制，但这里对sum_i还有下界限制L。

一种方法是用上下界处理，但可能出现负环，解决方法是初始做贪心来避免负权边。

另一种方法是，对于$i \in [k-L+1, k]$，将方程改为$sum_i = sum_{i-1} - x_i$，这样就将原来的$sum_i (i \in [k, n])$减去了L，于是只需让$sum_i \le R - L$即可。建出的图是DAG，因此没有负权环的问题。

CF 704D

平面上有n个球，现在要给这些球染色。一个球染成红色有r的花费，染成蓝色有b的花费。有两种类型的限制：

- 所有横坐标为l_i的球中，蓝球和红球个数的差的绝对值不超过d_i。
- 所有纵坐标为l_i的球中，蓝球和红球个数的差的绝对值不超过d_i。

给每个球染色，并在满足限制的前提下最小化花费。

$n, m \leq 10^5$

Solution

如果一个横坐标或纵坐标上有k个球，那么限制可以进行转化：

Solution

如果一个横坐标或纵坐标上有k个球，那么限制可以进行转化：

$$|red - blue| \leq d \Rightarrow \lceil \frac{k-d}{2} \rceil \leq red \leq \lfloor \frac{k+d}{2} \rfloor$$

Solution

如果一个横坐标或纵坐标上有k个球，那么限制可以进行转化：

$$|red - blue| \leq d \Rightarrow \lceil \frac{k-d}{2} \rceil \leq red \leq \lfloor \frac{k+d}{2} \rfloor$$

对每个横坐标建一个点，从源点向每个横坐标连流量上下界为red限制的边；纵坐标的点类似地向汇点连边。对于一个点(x, y)，从x向y连边，流量为1。

Solution

如果一个横坐标或纵坐标上有k个球，那么限制可以进行转化：

$$|red - blue| \leq d \Rightarrow \lceil \frac{k-d}{2} \rceil \leq red \leq \lfloor \frac{k+d}{2} \rfloor$$

对每个横坐标建一个点，从源点向每个横坐标连流量上下界为red限制的边；纵坐标的点类似地向汇点连边。对于一个点(x, y)，从x向y连边，流量为1。

如果$r > b$，那么我们要做的是最大可行流；否则要做的是最小可行流。

Solution

如果一个横坐标或纵坐标上有k个球，那么限制可以进行转化：

$$|red - blue| \leq d \Rightarrow \lceil \frac{k-d}{2} \rceil \leq red \leq \lfloor \frac{k+d}{2} \rfloor$$

对每个横坐标建一个点，从源点向每个横坐标连流量上下界为red限制的边；纵坐标的点类似地向汇点连边。对于一个点(x, y)，从x向y连边，流量为1。

如果$r > b$，那么我们要做的是最大可行流；否则要做的是最小可行流。

建出的图与二分图匹配类似，用Dinic可以做到$O(n\sqrt{n})$的复杂度。

CF 786E

有一棵n个节点的树，树上有m个居民，每天每个居民需要从x_i走到y_i去工作；此外，在每条边上还有一个守卫。

这里的居民很喜欢小狗。一个居民是高兴的，当且仅当这个居民拥有一只小狗，或者x_i到y_i路径上的所有守卫都拥有一只小狗。

一开始所有守卫或居民都没有小狗，求让所有居民高兴至少需要多少小狗。

$n \leq 2 \times 10^4, m \leq 10^4$

Solution

对所有居民和守卫建一个点，源点向居民连流量为1的边，守卫向汇点连流量为1的边；一个居民向其路径上的所有守卫连流量为∞的边。做最小割即可。

Solution

对所有居民和守卫建一个点，源点向居民连流量为1的边，守卫向汇点连流量为1的边；一个居民向其路径上的所有守卫连流量为∞的边。做最小割即可。

这样边数难以接受，考虑用倍增来优化建边。记辅助点$f_{i,j}$，表示从i到它的第2^j个祖先之间的所有守卫；然后从$f_{i,j}$到$f_{i,j-1}$和$f_{u,j-1}$连∞的边，其中u表示i的第2^{j-1}个祖先。

Solution

对所有居民和守卫建一个点，源点向居民连流量为1的边，守卫向汇点连流量为1的边；一个居民向其路径上的所有守卫连流量为∞的边。做最小割即可。

这样边数难以接受，考虑用倍增来优化建边。记辅助点$f_{i,j}$，表示从i到它的第2^j个祖先之间的所有守卫；然后从$f_{i,j}$到$f_{i,j-1}$和$f_{u,j-1}$连∞的边，其中u表示i的第2^{j-1}个祖先。

复杂度（据说）为$O((n+m)\sqrt{n+m}\log n)$。

ZJOI2010 贪吃的老鼠

奶酪店里有n块奶酪，第i块奶酪大小为sz_i，会在第l_i秒被生产出来，第r_i秒变质。有m只老鼠，第i只老鼠吃奶酪的速度为v_i，也就是说第i只老鼠单独吃掉第j块奶酪所需时间为sz_j/v_i。老鼠们吃奶酪有这样的限制：

- 一只老鼠不能同时吃两块奶酪
- 两只老鼠不能同时吃一块奶酪

老鼠们希望吃完所有的奶酪，为此它们需要使用魔法。具体来说，它们可以选定一个实数T，将所有r_i加上T。在能吃完奶酪的情况下最小化T。

$n, m \leq 30$

Solution

显然可以二分答案转化成判定性问题。

Solution

显然可以二分答案转化成判定性问题。

先不考虑“两只老鼠不能同时吃一块奶酪”的限制。以奶酪出现和变质的时间点为界，将时间划分成若干个时间段，设第i个时间段的长度为t_i。

Solution

显然可以二分答案转化成判定性问题。

先不考虑“两只老鼠不能同时吃一块奶酪”的限制。以奶酪出现和变质的时间点为界，将时间划分成若干个时间段，设第i个时间段的长度为t_i。

- 对于第i只老鼠的第j个时间段建一个点$v(i,j)$，源点向$v(i,j)$连流量为$v_i \times t_j$的边。

Solution

显然可以二分答案转化成判定性问题。

先不考虑“两只老鼠不能同时吃一块奶酪”的限制。以奶酪出现和变质的时间点为界，将时间划分成若干个时间段，设第i个时间段的长度为t_i。

- 对于第i只老鼠的第j个时间段建一个点$v(i,j)$，源点向$v(i,j)$连流量为$v_i \times t_j$的边。
- 对于每个奶酪建一个点，$v(i,j)$向所有第j个时间段可用的奶酪连$v_i \times t_j$的边，每个奶酪向汇点连sz_i的边。

Solution

显然可以二分答案转化成判定性问题。

先不考虑“两只老鼠不能同时吃一块奶酪”的限制。以奶酪出现和变质的时间点为界，将时间划分成若干个时间段，设第i个时间段的长度为t_i。

- 对于第i只老鼠的第j个时间段建一个点$v(i,j)$，源点向$v(i,j)$连流量为$v_i \times t_j$的边。
- 对于每个奶酪建一个点，$v(i,j)$向所有第j个时间段可用的奶酪连$v_i \times t_j$的边，每个奶酪向汇点连sz_i的边。

这样只要所有奶酪的边满流就有解。现在考虑怎么处理限制。

Solution

考虑一个奶酪j在一个时间段k内被吃的情况，设第i只老鼠吃的时间为e_i，那么这个奶酪被吃的部分S为：

Solution

考虑一个奶酪j在一个时间段k内被吃的情况，设第i只老鼠吃的时间为e_i，那么这个奶酪被吃的部分S为：

$$
\begin{aligned}
S = \sum_{i=1}^{m} e_i \times v_i &= \sum_{i=1}^{m} e_i \times (\sum_{j=1}^{i} v_j - v_{j-1}) \\
&= \sum_{j=1}^{m}(\sum_{i=j}^{n} e_i) \times (v_j - v_{j-1})
\end{aligned}
$$

Solution

考虑一个奶酪j在一个时间段k内被吃的情况，设第i只老鼠吃的时间为e_i，那么这个奶酪被吃的部分S为：

$$S = \sum_{i=1}^{m} e_i \times v_i = \sum_{i=1}^{m} e_i \times (\sum_{j=1}^{i} v_j - v_{j-1})$$
$$= \sum_{j=1}^{m} (\sum_{i=j}^{n} e_i) \times (v_j - v_{j-1})$$

那么将所有老鼠按v_j从小到大排序，然后令$v'_j = v_j - v_{j-1}$，直接按之前的方式连边即可。

Solution

考虑一个奶酪j在一个时间段k内被吃的情况，设第i只老鼠吃的时间为e_i，那么这个奶酪被吃的部分S为：

$$
\begin{aligned}
S &= \sum_{i=1}^{m} e_i \times v_i = \sum_{i=1}^{m} e_i \times (\sum_{j=1}^{i} v_j - v_{j-1}) \\
&= \sum_{j=1}^{m} (\sum_{i=j}^{n} e_i) \times (v_j - v_{j-1})
\end{aligned}
$$

那么将所有老鼠按v_j从小到大排序，然后令$v'_j = v_j - v_{j-1}$，直接按之前的方式连边即可。

原来的问题是只能保证$e_i \le t_k$而无法保证$\sum e_i \le t_k$，而对v做差分后，$e'_j = \sum_{i=j}^{n} e_j$，$e'_1$就保证了限制。

Shortest Path

- 差分约束系统：有若干个变量和许多形如$x_i - x_j \leq w_k$的限制，求一组解。做法是从j到i连边w_k，然后做最短路；如果图中出现了负权环则无解。

CF 986F

有一排n个人，每一轮中第i个位置上的人会走到P_i（$P_i \neq i$）。给出n, k，判断是否存在P，使得k轮后所有人站在原来的位置上。

有多组询问，但所有询问中不同的k不会超过50个。

$n \leq 10^{18}, k \leq 10^{15}, T \leq 10^4$。

Solution

由于k轮后所有人会到原位，所以P是排列且P的所有环长都是k的约数，我们需要判断用k的约数能否凑出n。

Solution

由于k轮后所有人会到原位，所以P是排列且P的所有环长都是k的约数，我们需要判断用k的约数能否凑出n。

不难发现只需用k的质因子即可.设有c个质因子，那么问题变为给出c个质因子p_i，判断$\sum_{i=1}^{c} p_i x_i = n$是否有正整数解。

Solution

由于k轮后所有人会到原位，所以P是排列且P的所有环长都是k的约数，我们需要判断用k的约数能否凑出n。

不难发现只需用k的质因子即可.设有c个质因子，那么问题变为给出c个质因子p_i，判断$\sum_{i=1}^{c} p_i x_i = n$是否有正整数解。

- $c = 1$，直接判断k能否整除n.

Solution

由于k轮后所有人会到原位，所以P是排列且P的所有环长都是k的约数，我们需要判断用k的约数能否凑出n。

不难发现只需用k的质因子即可.设有c个质因子，那么问题变为给出c个质因子p_i，判断$\sum_{i=1}^{c} p_i x_i = n$是否有正整数解。

- $c = 1$，直接判断k能否整除n.
- $c = 2$，用拓展欧几里得算出一组整数解，再通过调整判断是否有正整数解。

Solution

$c > 2$时，此时最小的质因子p_1必然小于10^5。注意到如果$n = x$时有解，那么$n = x + p_1$时也有解。

Solution

$c > 2$时，此时最小的质因子p_1必然小于10^5。注意到如果$n = x$时有解，那么$n = x + p_1$时也有解。

于是可以定义$dis(i)$表示最小的x，满足$n = x$时有解，且$x \equiv i \pmod{p_1}$。将加上一个质因子的操作看成边，用最短路计算dis即可。

Solution

$c > 2$时，此时最小的质因子p_1必然小于10^5。注意到如果$n = x$时有解，那么$n = x + p_1$时也有解。

于是可以定义$dis(i)$表示最小的x，满足$n = x$时有解，且$x \equiv i \pmod{p_1}$。将加上一个质因子的操作看成边，用最短路计算dis即可。

对于一个n，只需判断$dis(n \mod p_1)$与n的大小关系。$O(50 \times k^{\frac{1}{3}} \log k)$

TC SRM553 Div1 Hard/51NOD 1340

有一个地铁环线，环线中有n个站台，标号为$0 \to n-1$。这个环线是单行线，从i到$(i+1) \mod n$有一条有向边，每条边的长度是一个正整数。有m个限制，每个限制可能是$dis(i,j) \le w$或者$dis(i,j) \ge w$。

求地铁的总长有多少种可能，如果有无限种可能则输出-1。

$n \le 50, w \le 10^9$

Solution1

不难发现是差分约束的模型。设$s_i = dis(0, i)$，总长为x，那么对于限制$dis(i, j) \leq w$则可列出不等式：

Solution1

不难发现是差分约束的模型。设$s_i = dis(0,i)$，总长为x，那么对于限制$dis(i,j) \leq w$则可列出不等式：

- $s_j - s_i \leq w, i < j$.
- $s_j - s_i \leq w - x, j < i$.

Solution1

不难发现是差分约束的模型。设$s_i = dis(0,i)$, 总长为x, 那么对于限制$dis(i,j) \le w$则可列出不等式：

- $s_j - s_i \le w, i < j$.
- $s_j - s_i \le w - x, j < i$.

对于另一类不等式，只需两边取负号即可转化成类似的形式。由于边的长度为正整数，还需加上限制:

Solution1

不难发现是差分约束的模型。设$s_i = dis(0,i)$，总长为x，那么对于限制$dis(i,j) \le w$则可列出不等式：

- $s_j - s_i \le w, i < j$.
- $s_j - s_i \le w - x, j < i$.

对于另一类不等式，只需两边取负号即可转化成类似的形式。由于边的长度为正整数，还需加上限制:

- $dis(i, i+1) \ge 1$.
- $dis(0, n-1) \le x - 1$。

Solution1

这里的边权中有未知数x，不过其系数在$[-1,1]$之间。我们只需要知道图中是否有负环，而每个环的长度也可以表示成$kx+b$的形式，且k在$[-n,n]$之间。

Solution1

这里的边权中有未知数x，不过其系数在$[-1,1]$之间。我们只需要知道图中是否有负环，而每个环的长度也可以表示成$kx+b$的形式，且k在$[-n,n]$之间。

枚举环上的一个点u，记$dis(i,k)$表示，从u到i的路径，其长度可以表示为$kx+b$，这些路径中b最小是多少。这可以在$O(n^2(n+m))$的时间内计算。

Solution1

这里的边权中有未知数x，不过其系数在$[-1,1]$之间。我们只需要知道图中是否有负环，而每个环的长度也可以表示成$kx+b$的形式，且k在$[-n,n]$之间。

枚举环上的一个点u，记$dis(i,k)$表示，从u到i的路径，其长度可以表示为$kx+b$，这些路径中b最小是多少。这可以在$O(n^2(n+m))$的时间内计算。

这样我们可以得到对于每个k，b最小的环的长度，找到使这些环长均非负的区间即可。$O(n^3(n+m))$。

Solution1

这里的边权中有未知数x，不过其系数在$[-1,1]$之间。我们只需要知道图中是否有负环，而每个环的长度也可以表示成$kx+b$的形式，且k在$[-n,n]$之间。

枚举环上的一个点u，记$dis(i,k)$表示，从u到i的路径，其长度可以表示为$kx+b$，这些路径中b最小是多少。这可以在$O(n^2(n+m))$的时间内计算。

这样我们可以得到对于每个k，b最小的环的长度，找到使这些环长均非负的区间即可。$O(n^3(n+m))$。

这个做法已经足以通过了，但还有一个更快(shi)的做法。

Solution2

对所有边进行$n-1$次松弛之后，如果仍存在可以松弛的边（即$d(v) > d(u) + w(u,v)$）那么图中存在负权环。

Solution2

对所有边进行$n-1$次松弛之后，如果仍存在可以松弛的边（即$d(v) > d(u) + w(u,v)$）那么图中存在负权环。

建图是一样的，但我们不再枚举环上的点，而是直接从0号点做最短路，同样地计算$dis(i,k)$。

Solution2

对所有边进行$n-1$次松弛之后，如果仍存在可以松弛的边（即$d(v) > d(u) + w(u, v)$）那么图中存在负权环。

建图是一样的，但我们不再枚举环上的点，而是直接从0号点做最短路，同样地计算$dis(i, k)$。

这样我们可以将$d(i)$表示为一些一次函数的最小值（即$d(i) = \min_{k=-n}^{n}(k \times x + dis(i, k))$），对这些一次函数求上凸壳即可得到$d(i)$的分段函数。

Solution2

对所有边进行$n-1$次松弛之后，如果仍存在可以松弛的边（即$d(v) > d(u) + w(u,v)$）那么图中存在负权环。

建图是一样的，但我们不再枚举环上的点，而是直接从0号点做最短路，同样地计算$dis(i,k)$。

这样我们可以将$d(i)$表示为一些一次函数的最小值（即$d(i) = \min_{k=-n}^{n}(k \times x + dis(i,k))$），对这些一次函数求上凸壳即可得到$d(i)$的分段函数。

求出这些信息后，枚举每条边(u,v)，不难算出x在哪些区间内时这条边仍可以松弛。对所有这样的区间求并即得到不合法区间。$O(n^2(n+m))$。

Connectivity

- 传递闭包：其实就是对一个有向图，对每个点求这个点能到哪些点。先求强连通分量，然后dp即可，可用bitset优化。

Connectivity

- 传递闭包：其实就是对一个有向图，对每个点求这个点能到哪些点。先求强连通分量，然后dp即可，可用bitset优化。
- 支配树：在有向图中，给定起点s。对于一个从s可达的点u，称另一个点v支配u，当且仅当删去v后s到u不可达。定义u的最近支配点$idom(u)$为所有支配u的点中距离最近的点。将$idom(u)$看做u的父亲，则形成了一个树结构。

CF 757F

给出一张带权有向图和起点S，选择一个除S以外的点，使得删掉这个点后$dis(S,u)$变大的u尽量多。

$n \le 2 \times 10^5$

Solution

求出以S为根的最短路DAG（如果$dis(v) = dis(u) + w(u, v)$，那么边$(u, v)$在最短路DAG中）。

Solution

求出以S为根的最短路DAG（如果$dis(v)=dis(u)+w(u,v)$，那么边(u,v)在最短路DAG中）。

可以发现在这个DAG上，如果u能支配v，那么在原图上删去u后v的最短路变大。只要求出支配树就不难计算答案。

Solution

求出以S为根的最短路DAG（如果$dis(v) = dis(u) + w(u, v)$，那么边$(u, v)$在最短路DAG中）。

可以发现在这个DAG上，如果u能支配v，那么在原图上删去u后v的最短路变大。只要求出支配树就不难计算答案。

DAG上的支配树非常好求。对于一个点u，设其入边为$(p_1, u), (p_2, u), ..., (p_k, u)$，则$idom(u)$为支配树上$p_1, p_2, ..., p_k$的$lca$。

$O((n + m) \log n)$

CF 295E

一个无向图，你需要找两个点s, t，使得s到t有三条不相交的简单路径。判断是否有解并输出方案。

$n, m \leq 2 \times 10^5$

Solution

首先s和t一定在一个点双内，因此可以先求出点双，然后每个点双分别判断。

Solution

首先s和t一定在一个点双内，因此可以先求出点双，然后每个点双分别判断。

对于一个点双，如果这个点双是一条边，或者整个点双是一个简单环，那么这个点双中是无解的；否则一定有解。

Solution

首先s和t一定在一个点双内，因此可以先求出点双，然后每个点双分别判断。

对于一个点双，如果这个点双是一条边，或者整个点双是一个简单环，那么这个点双中是无解的；否则一定有解。

考虑怎么构造，首先找到任意一个环C。然后在环上找一个点u，满足存在一条边(u, v)，这条边不是环上的边，并且v在这个点双内。从v一直往下dfs，一定可以dfs到一个除u以外的环上的点。取出这段dfs出的路径，加上之前的找出的环C，不难构造出方案。$O(n + m)$。

NEERC2016 B

给出n个0/1串，每个0/1串中至多有一个'?'。问是否存在一种在'?'处填0/1的方案，使得n个串中，不存在一个串是另一个串的前缀。输出方案。

$n, \sum |S| \leq 5 \times 10^5$

Solution

不难发现是一个2-sat模型，可以直接枚举每两个串判断是否冲突进行连边，但这样是平方的。

Solution

不难发现是一个2-sat模型，可以直接枚举每两个串判断是否冲突进行连边，但这样是平方的。

由于限制和前缀有关，可以想到用Trie来优化建边。

Solution

不难发现是一个2-sat模型，可以直接枚举每两个串判断是否冲突进行连边，但这样是平方的。

由于限制和前缀有关，可以想到用Trie来优化建边。

一种做法是，Trie上每个节点u记一个辅助变量f_u，表示这个子树中是否有串被选择。把串按长度从大到小加入Trie，并将Trie可持久化，连边比较简单，只需保证加入的这个串与之前的串没有冲突即可。

Solution

不难发现是一个2-sat模型，可以直接枚举每两个串判断是否冲突进行连边，但这样是平方的。

由于限制和前缀有关，可以想到用Trie来优化建边。

一种做法是，Trie上每个节点u记一个辅助变量f_u，表示这个子树中是否有串被选择。把串按长度从大到小加入Trie，并将Trie可持久化，连边比较简单，只需保证加入的这个串与之前的串没有冲突即可。

点数边数均为线性，$O(n + |S|)$

Thanks