图论及其相关例题

ylsoi

雅礼中学

October 26, 2018

Preface

pym上午讲的内容应已经很多了

Preface

pym上午讲的内容应已经很多了

下午主要讲一些题目

Preface

pym上午讲的内容应已经很多了

下午主要讲一些题目

做过的同学可以随意上来秒题,带一带节奏...

关于最短路

SPFA,Dijkstra,Floyd什么的都已经讲烂了，一般难的都是怎么建模什么的。

关于最短路

SPFA,Dijkstra,Floyd什么的都已经讲烂了，一般难的都是怎么建模什么的。

联赛的时候要注意选择**复杂度优秀**的最短路算法，如果有多组数据，数组需要记得清零

关于最短路

SPFA,Dijkstra,Floyd什么的都已经讲烂了，一般难的都是怎么建模什么的。

联赛的时候要注意选择**复杂度优秀**的最短路算法，如果有多组数据，数组需要记得清零

这里就直接放几道题了。

墨墨的等式

墨墨突然对等式很感兴趣，他正在研究$a_1x_1 + a_2x_2 + \ldots + a_nx_n = B$存在非负整数解的条件

他要求你编写一个程序，给定$N, \{a_n\}$以及B的取值范围，求出有多少B可以使等式存在非负整数解。

$N \leq 12, 0 \leq a_i \leq 5 \times 10^5, 1 \leq B_{min} \leq B_{max} \leq 10^{12}$

分析

这个题目貌似非常经典了。

分析

这个题目貌似非常经典了。题目中要求的是非负整数解，于是我们可以把每一个数看成一个物品，求所有物品可以组成的体积。

分析

这个题目貌似非常经典了。题目中要求的是非负整数解，于是我们可以把每一个数看成一个物品，求所有物品可以组成的体积。

但是直接跑背包显然是接受不了的

分析

这个题目貌似非常经典了。题目中要求的是非负整数解，于是我们可以把每一个数看成一个物品，求所有物品可以组成的体积。

但是直接跑背包显然是接受不了的

考虑到最后的体积集合，我们把它按照a[1]（也就是任意一个数）的剩余类分类。

分析

这个题目貌似非常经典了。题目中要求的是非负整数解，于是我们可以把每一个数看成一个物品，求所有物品可以组成的体积。

但是直接跑背包显然是接受不了的

考虑到最后的体积集合，我们把它按照a[1]（也就是任意一个数）的剩余类分类。

虽然可以组成的体积种类很多，但是按照剩余类分类之后体积的种类就只有5e5种

分析

这个题目貌似非常经典了。题目中要求的是非负整数解，于是我们可以把每一个数看成一个物品，求所有物品可以组成的体积。

但是直接跑背包显然是接受不了的

考虑到最后的体积集合，我们把它按照a[1]（也就是任意一个数）的剩余类分类。

虽然可以组成的体积种类很多，但是按照剩余类分类之后体积的种类就只有5e5种

于是我们只需要求出每一个剩余类最小的体积就好了,每一个合法的体积一定可以表示成$a[1] \times x + b$。

建图

建图其实也很简单。

每一个余数代表一个点，点u可以向点$(v+a[j]) \mod a[1]$连一条权值为a[j]的边。

建图

建图其实也很简单。

每一个余数代表一个点，点u可以向点$(v + a[j]) \mod a[1]$连一条权值为a[j]的边。

从0开始单源最短路即可以得到所有余数的最小体积。

故乡的梦

给定一个带权无向图，每次询问删除一条边之后从S到T的最短路是多少？（各个询问之间独立）

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

如果删除的边不在最短路中或者可以被替换，那么答案即为最短路。

如果删除的边不在最短路中或者可以被替换，那么答案即为最短路。

如果删除的边在最短路中并且不可以被替换，考虑将这条边删除的新图：

假设原来的最短路为$S->T$，那么新的最短路一定是$S->u->x->y->v->T$（其中u，v在原来的最短路上，x，y不在原来的最短路上）

如果删除的边不在最短路中或者可以被替换，那么答案即为最短路。

如果删除的边在最短路中并且不可以被替换，考虑将这条边删除的新图：

假设原来的最短路为$S->T$，那么新的最短路一定是$S->u->x->y->v->T$（其中u，v在原来的最短路上，x，y不在原来的最短路上）

于是我们发现如果删除了这条边之后S和T联通，那么一定会经过形如$x->y$这样的路径，即经过原本不在最短路中的边。

并且对于$x->y$中的一条边(uu,vv)，$S->uu$和$vv->T$一定是两条最短的路径

并且对于$x->y$中的一条边(uu,vv)，$S->uu$和$vv->T$一定是两条最短的路径

于是我们枚举出每一条不在最短路中的边(uu,vv)，并计算出对应的最早的u和最晚的v，用强制经过(uu,vv)的最短路去更新u和v中间缺失的那些边的答案。

并且对于$x->y$中的一条边(uu,vv)，$S->uu$和$vv->T$一定是两条最短的路径

于是我们枚举出每一条不在最短路中的边(uu,vv)，并计算出对应的最早的u和最晚的v，用强制经过(uu,vv)的最短路去更新u和v中间缺失的那些边的答案。

为了方便，在具体实现的时候随意提出一条$S->T$的最短路并把它当作唯一的最短路即可。

探险

给定一个无向图，每一条边正着和反着都有一个边权，求一条不经过重复边的路径，使得边权和最小。

$n \leq 10^4, m \leq 10^5$

分析

这个题目的思路比较巧妙~~根本想不到~~

分析

这个题目的思路比较巧妙~~根本想不到~~

网络上的题解只有做法，没有详细地解释。

分析

这个题目的思路比较巧妙~~根本想不到~~

网络上的题解只有做法，没有详细地解释。

考虑最暴力的方法，对于1号点能够直接到达的每一个点,把它和1号点的边删掉以后,以这个点为源点跑最短路。

分析

这个题目的思路比较巧妙~~根本想不到~~

网络上的题解只有做法，没有详细地解释。

考虑最暴力的方法，对于1号点能够直接到达的每一个点,把它和1号点的边删掉以后,以这个点为源点跑最短路。

但是这样会T飞。

分析

不妨先放下不能重复走的限制，建立一个新的虚拟汇点t=n+1，然后将所有连向1的边转化成连向t的边。然后对于1到t跑最短路算法

分析

不妨先放下不能重复走的限制，建立一个新的虚拟汇点t=n+1，然后将所有连向1的边转化成连向t的边。然后对于1到t跑最短路算法

这样子做显然有一些点问题，有的路径可能刚刚才走出去就又回来了，这样是不合法的。

分析

不妨先放下不能重复走的限制，建立一个新的虚拟汇点t=n+1，然后将所有连向1的边转化成连向t的边。然后对于1到t跑最短路算法

这样子做显然有一些点问题，有的路径可能刚刚才走出去就又回来了，这样是不合法的。

例如一条边(1,v,w),在转移中可能会是这个样子的 $1->v->t$

分析

不妨先放下不能重复走的限制，建立一个新的虚拟汇点t=n+1，然后将所有连向1的边转化成连向t的边。然后对于1到t跑最短路算法

这样子做显然有一些点问题，有的路径可能刚刚才走出去就又回来了，这样是不合法的。

例如一条边(1,v,w),在转移中可能会是这个样子的 $1->v->t$

那么如果我们将(1,v,w)给去掉，将可能从(1,v,w)这条边出发的路径转化为一些边(1,u,dis[1...u]) (1,v,w)$\in$ this path

分析

不妨先放下不能重复走的限制，建立一个新的虚拟汇点t=n+1，然后将所有连向1的边转化成连向t的边。然后对于1到t跑最短路算法

这样子做显然有一些点问题，有的路径可能刚刚才走出去就又回来了，这样是不合法的。

例如一条边(1,v,w),在转移中可能会是这个样子的 $1->v->t$

那么如果我们将(1,v,w)给去掉，将可能从(1,v,w)这条边出发的路径转化为一些边(1,u,dis[1...u]) (1,v,w)$\in$ this path

使得这些u存在于所有$1->t$的路径上，并且对于所有的u,都有1到u的最短路的第一个点不是v。

分析

这样以后我们再去求最短路，便不用担心一条边跑出去再跑回去的问题，因为有了上面的限制，通过这条边出去之后再回去一定不是最短的。

分析

这样以后我们再去求最短路，便不用担心一条边跑出去再跑回去的问题，因为有了上面的限制，通过这条边出去之后再回去一定不是最短的。

考虑怎么建立上面所说的等效边，可以对于原图做一次最短路，记录每个点的最短路径的第一个点f[u]，枚举每一条边(u,v,w)，如果f[u]!=f[v]，那么则说明这是f[u]和f[v]的范围的分界处，在这里分别对f[u]和f[v]都建立等效边即可。

分析

这样以后我们再去求最短路，便不用担心一条边跑出去再跑回去的问题，因为有了上面的限制，通过这条边出去之后再回去一定不是最短的。

考虑怎么建立上面所说的等效边，可以对于原图做一次最短路，记录每个点的最短路径的第一个点f[u]，枚举每一条边(u,v,w)，如果f[u]!=f[v]，那么则说明这是f[u]和f[v]的范围的分界处，在这里分别对f[u]和f[v]都建立等效边即可。

当然如果u=1或者v=1的情况需要特殊考虑，但是方法是类似的。具体方法可以参考hzwer的博客。

最小生成树一般使用Kruskal或者Prim算法求解。

每一种算法都有自己的优点，这里不再赘述。

最小生成树也有一些优秀的性质，接下来就直接来看题吧。。。

给定一些点，每一个点有编号和权值，求权值和最大的点集使得点集中不存在编号异或和为0的子集

$N \leq 1000, id \leq 10^{18}, val \leq 10^4$

其实这一题和MST的关系并不是很大，只是可以用类似的贪心的思想来解决。

其实这一题和MST的关系并不是很大，只是可以用类似的贪心的思想来解决。

首先我们可以把编号为0的点特判掉

其实这一题和MST的关系并不是很大，只是可以用类似的贪心的思想来解决。

首先我们可以把编号为0的点特判掉

这时，我们可以把一个异或和为0的集合看成是一个环，这个环上的所有数字异或起来为0，只要我们将其中任意一条边删除，便不存在异或和为0的环了。

其实这一题和MST的关系并不是很大，只是可以用类似的贪心的思想来解决。

首先我们可以把编号为0的点特判掉

这时，我们可以把一个异或和为0的集合看成是一个环，这个环上的所有数字异或起来为0，只要我们将其中任意一条边删除，便不存在异或和为0的环了。

抽象一下我们便可以把这道题用Kruskal的贪心去做，按照权值大小排序之后依次将每个点加入，如果此时集合中存在异或和等于这个数的子集，则不加入这个数。

其实这一题和MST的关系并不是很大，只是可以用类似的贪心的思想来解决。

首先我们可以把编号为0的点特判掉

这时，我们可以把一个异或和为0的集合看成是一个环，这个环上的所有数字异或起来为0，只要我们将其中任意一条边删除，便不存在异或和为0的环了。

抽象一下我们便可以把这道题用Kruskal的贪心去做，按照权值大小排序之后依次将每个点加入，如果此时集合中存在异或和等于这个数的子集，则不加入这个数。

加入的过程用线性基来维护，最后加入集合的数的总和就是答案。

Petrol

给定一个n个点、m条边的带权无向图，其中有s个点是加油站。

每辆车都有一个油量上限b，即每次行走距离不能超过b，但在加油站可以补满。

q次询问，每次给出x,y,b，表示出发点是x，终点是y，油量上限为b，且保证x点和y点都是加油站，请回答能否从x走到y。

$2 \leq s \leq n \leq 200000, 1 \leq m \leq 200000, 1 \leq q \leq 200000$

分析

不难发现如果要顺利地完成旅程，一定是从一个加油站跑到另外一个加油站去，并且任意两个加油站之间地距离不可以超过b。

分析

不难发现如果要顺利地完成旅程，一定是从一个加油站跑到另外一个加油站去，并且任意两个加油站之间地距离不可以超过b。

于是便转化成了这样一个问题：加油站为关键点，求一条路径使得两两关键点之间地路径长度的最大值最小

分析

不难发现如果要顺利地完成旅程，一定是从一个加油站跑到另外一个加油站去，并且任意两个加油站之间地距离不可以超过b。

于是便转化成了这样一个问题：加油站为关键点，求一条路径使得两两关键点之间地路径长度的最大值最小

这个时候最小生成树的性质便起到了作用，将两两加油站之间的最短距离计算出来，并且只对关键点求解最小生成树，生成树上的路径一定是最优方案，具体地证明可以参照Kruskal算法的运行过程。

分析

显然关键点的个数太多了，如果我们两两关键点之间求解最短路会T。。

分析

显然关键点的个数太多了，如果我们两两关键点之间求解最短路会T。。

此时就要考虑最小生成树的性质，最大化地减少连接无用的边。

分析

显然关键点的个数太多了，如果我们两两关键点之间求解最短路会T。。

此时就要考虑最小生成树的性质，最大化地减少连接无用的边。

考虑关键点s到关键点t的路径，若它们中途经过了另外一个关键点u，那么$s->u, u->t$一定会更优。

分析

显然关键点的个数太多了，如果我们两两关键点之间求解最短路会T。。

此时就要考虑最小生成树的性质，最大化地减少连接无用的边。

考虑关键点s到关键点t的路径，若它们中途经过了另外一个关键点u，那么$s->u, u->t$一定会更优。

如果s到t的路径可以被$s->a1, a1->a2...an->t$,这样的路径所取代，并且其中任意一条路径的长度均小于$s->t$，那么$s->t$还是不要选择的好。

分析

于是我们可以对于每一个点求出离它最近的关键点，记为f[u]，如果一条边(u,v)的f[u],f[v]不同，那么则将(f[u],f[v])加入候选边。

分析

于是我们可以对于每一个点求出离它最近的关键点，记为f[u]，如果一条边(u,v)的f[u],f[v]不同，那么则将(f[u],f[v])加入候选边。

如果一条关键点之间的路径不可以被上述方法得到，那么它肯定不能加入最小生成树中，即不满足上面的几个条件。

分析

于是我们可以对于每一个点求出离它最近的关键点，记为f[u]，如果一条边(u,v)的f[u],f[v]不同，那么则将(f[u],f[v])加入候选边。

如果一条关键点之间的路径不可以被上述方法得到，那么它肯定不能加入最小生成树中，即不满足上面的几个条件。

即这条不符合条件的路径，要么路径中有其它的关键点，要么可以被一条其它的路径取代。

分析

于是我们可以对于每一个点求出离它最近的关键点，记为f[u]，如果一条边(u,v)的f[u],f[v]不同，那么则将(f[u],f[v])加入候选边。

如果一条关键点之间的路径不可以被上述方法得到，那么它肯定不能加入最小生成树中，即不满足上面的几个条件。

即这条不符合条件的路径，要么路径中有其它的关键点，要么可以被一条其它的路径取代。

具体地实现可以将所有关键点加入堆中跑Djkstra，然后对于所有询问离线即可。

Tarjan

Tarjan 发明的算法很多很多，这里只有图论上的Tarjan的题目。

Quare

给定一个图，求一个边的子集，使得整张图为边双连通并且边的权值和最小。

$n \leq 12, m \leq 40$

分析

数据范围这么小，考虑状压DP。

分析

数据范围这么小，考虑状压DP。

题目要求子图为边双连通，边双连通可以表示成若干个环套在一起，但是这样并不方便我们表示状态。

分析

数据范围这么小，考虑状压DP。

题目要求子图为边双连通，边双连通可以表示成若干个环套在一起，但是这样并不方便我们表示状态。

思考一下，不难发现一个边双可以这样组成：一个边双不断地添加一条链并且使这条链首尾都和边双里的任意一个点相连。

分析

数据范围这么小，考虑状压DP。

题目要求子图为边双连通，边双连通可以表示成若干个环套在一起，但是这样并不方便我们表示状态。

思考一下，不难发现一个边双可以这样组成：一个边双不断地添加一条链并且使这条链首尾都和边双里的任意一个点相连。

于是转移的大致思路便出来了，枚举一条链并且将这一条链接入目前的集合中，每一次保证最小代价。

分析

于是我们要预处理任意一条链自己本身连通的最小值

分析

于是我们要预处理任意一条链自己本身连通的最小值

还有一个点向一个集合中连边的最小值

分析

于是我们要预处理任意一条链自己本身连通的最小值

还有一个点向一个集合中连边的最小值

由于一条链可能只有一个点，所以还要处理次小值

分析

于是我们要预处理任意一条链自己本身连通的最小值

还有一个点向一个集合中连边的最小值

由于一条链可能只有一个点，所以还要处理次小值

然后直接转移就好了。

灾难

草原中有一张食物网，如果将一种动物弄死，将会有一些动物没有食物而死亡，那么称死亡动物的种类数为这种动物的灾难值。

求每一种动物的灾难值。

$n \leq 65534$，输入数据$\leq$10M。

分析

题目可以转化为将DAG删除一个结点后有多少个结点变得不可达

分析

题目可以转化为将DAG删除一个结点后有多少个结点变得不可达

不难发现，整张食物网其实是构成了一个树形结构，即直接导致某种动物死亡的动物有且只有一种。某一个结点的死亡必定会导致它的子树内的所有的动物都死亡。

分析

题目可以转化为将DAG删除一个结点后有多少个结点变得不可达

不难发现，整张食物网其实是构成了一个树形结构，即直接导致某种动物死亡的动物有且只有一种。某一个结点的死亡必定会导致它的子树内的所有的动物都死亡。

于是我们尝试将这颗“灭绝树”给建出来，一种动物v要死亡，那么直接连向它的所有结点$u_1, u_2, u_3 \ldots$都必须要死亡。那么我们只要找到第一个可以使u全部死亡的结点，它就是v的父亲。

分析

题目可以转化为将DAG删除一个结点后有多少个结点变得不可达

不难发现，整张食物网其实是构成了一个树形结构，即直接导致某种动物死亡的动物有且只有一种。某一个结点的死亡必定会导致它的子树内的所有的动物都死亡。

于是我们尝试将这颗“灭绝树”给建出来，一种动物v要死亡，那么直接连向它的所有结点$u_1, u_2, u_3 \ldots$都必须要死亡。那么我们只要找到第一个可以使u全部死亡的结点，它就是v的父亲。

u必须在v之前处理出来，并且加入灭绝树中，于是直接按照拓扑序建树。之后的找父亲就是一个求lca的过程，只需要每次更新新结点的st表即可。

给定一个N个点M条边的有向无环图，每条边长度都是1。

请找到一个点，使得删掉这个点后剩余的图中的最长路径最短。

n$\leq 10^5$,m$\leq 10^6$。

分析

这个题目讲起来会比较混乱，至少我觉得思路不是很自然。

分析

这个题目讲起来会比较混乱，至少我觉得思路不是很自然。

首先我们加一个超级源S和一个超级汇T，然后整个题目就变成了求$S->T$的最长链。

分析

这个题目讲起来会比较混乱，至少我觉得思路不是很自然。

首先我们加一个超级源S和一个超级汇T，然后整个题目就变成了求$S->T$的最长链。

计算出S到每一个点的最长路和每一个点到T的最长路，这样我们就可以很方便地算出来经过任意一条边的最长路了。

分析

这个题目讲起来会比较混乱，至少我觉得思路不是很自然。

首先我们加一个超级源S和一个超级汇T，然后整个题目就变成了求$S->T$的最长链。

计算出S到每一个点的最长路和每一个点到T的最长路，这样我们就可以很方便地算出来经过任意一条边的最长路了。

考虑到删除一个点之后新的最长链一定会经过一些边，于是我们可以维护一个集合代表删除了这个点之后新的最长路可能会经过哪些边。

继续分析

假设我们删除了点u。

继续分析

假设我们删除了点u。

首先我们所维护的集合内的边一定和点u没有偏序关系，否则经过这条边的最长路可能会经过点u。

继续分析

假设我们删除了点u。

首先我们所维护的集合内的边一定和点u没有偏序关系，否则经过这条边的最长路可能会经过点u。

其次有偏序关系的边也没有必要重复放入集合中。

继续分析

假设我们删除了点u。

首先我们所维护的集合内的边一定和点u没有偏序关系，否则经过这条边的最长路可能会经过点u。

其次有偏序关系的边也没有必要重复放入集合中。

于是一个集合我们可以看成是一个割，其中的边都是平行的，不难发现图中的所有路径都经过了这个割中的边。删除一个点之后我们将这个割中和这个点相连的所有边去掉，剩下的边集的答案便一定是去掉了这个点之后的最长路的长度。

继续继续分析

于是接下来就只需要动态维护这个割集并使它满足上面的条件，按照拓扑序的过程，维护一个指向目前的点集的边集，每次选定一个点将所有指向它的边删除，这个集合便一定满足性质，计算完之后再将u删除，将所有从u出发的边加入集合即可。

继续继续分析

于是接下来就只需要动态维护这个割集并使它满足上面的条件，按照拓扑序的过程，维护一个指向目前的点集的边集，每次选定一个点将所有指向它的边删除，这个集合便一定满足性质，计算完之后再将u删除，将所有从u出发的边加入集合即可。

上述的过程可以用权值线段树或堆来实现。

狡猾的hany01

Hack国国王FKB很套路，他为了自己的套路，时不时想看一下机房其他的人在做什么题目，最近他又盯上了Hany01，想知道他的做题情况，于是你在他的威逼利诱下，接受了这个任务。

但是hany01也很套路，它会伪造自己的做题情况，你会假装成GKK去接近hany01,然后去偷偷瞄一眼它的做题记录。

hany01有一个记录表，你会趁hany01不在时去偷看，可是你无法将记录偷出来，每次偷看记录时你都只能看某段时间内记录的做题情况，并且你只能记住这段时间内的总题数。 现在，你总共偷看了m次记录，当然也就记住了m段时间内的做题数量 ，你的任务是根据记住的这些信息来判断hany01的做题数量是不是假的。

一道差分约束系统的裸题

一道差分约束系统的裸题

对于一个限制[l,r]，我们可以将它拆成$sum[r]-sum[l-1]\leq x, sum[r]-sum[l-1]\geq x$。

一道差分约束系统的裸题

对于一个限制[l,r]，我们可以将它拆成$sum[r]-sum[l-1]\leq x, sum[r]-sum[l-1]\geq x$。

然后然后将这个转化为差分约束系统的建图，如果不合法，则必定有$sum[p]-sum[p]\neq 0$，在环中正着走一遍和反着走一遍一定有一遍出现了负权环。

一道差分约束系统的裸题

对于一个限制[l,r]，我们可以将它拆成$sum[r]-sum[l-1]\leq x, sum[r]-sum[l-1]\geq x$。

然后然后将这个转化为差分约束系统的建图，如果不合法，则必定有$sum[p]-sum[p]\neq 0$，在环中正着走一遍和反着走一遍一定有一遍出现了负权环。

数据范围很小，直接**SPFA**判断负权环即可。

一道差分约束系统的裸题

对于一个限制[l,r]，我们可以将它拆成$sum[r]-sum[l-1]\leq x, sum[r]-sum[l-1]\geq x$。

然后然后将这个转化为差分约束系统的建图，如果不合法，则必定有$sum[p]-sum[p]\neq 0$，在环中正着走一遍和反着走一遍一定有一遍出现了负权环。

数据范围很小，直接**SPFA**判断负权环即可。

当然带权并查集也是可以做的。

THANKS!