差分约束，二分匹配， 2-sat

差分约束

差分约束系统 是一种特殊的 n 元一次不等式组，它包含 n 个变量 $x_1, x_2, ..., x_n$ 以及 m 个约束条件，每个约束条件是由两个其中的变量做差构成的，形如 $x_i - x_j \leq c_k$ ，其中 c_k 是常数（可以是非负数，也可以是负数）。我们要解决的问题是：求一组解 $x_1 = a_1, x_2 = a_2, ..., x_n = a_n$ ，使得所有的约束条件得到满足，否则判断出无解。

差分约束系统中的每个约束条件 $x_i - x_j \leq c_k$ 都可以变形成 $x_i \leq x_j + c_k$ ，这与单源最短路中的三角形不等式 $dist[y] \leq dist[x] + z$ 非常相似。因此，我们可以把每个变量 x_i 看做图中的一个结点，对于每个约束条件 $x_i - x_j \leq c_k$ ，从结点 j 向结点 i 连一条长度为 c_k 的有向边。

注意到，如果 $\{a_1, a_2, ..., a_n\}$ 是该差分约束系统的一组解，那么对于任意的常数 d ，$\{a_1 + d, a_2 + d, ..., a_n + d\}$ 显然也是该差分约束系统的一组解，因为这样做差后 d 刚好被消掉。

设 $dist[0]=0$ 并向每一个点连一条边，跑单源最短路，若图中存在负环，则给定的差分约束系统无解，否则， $x_i=dist[i]$ 为该差分约束系统的一组解。

一般使用 Bellman-Ford 或队列优化的 Bellman-Ford（俗称 SPFA，在某些随机图跑得很快）判断图中是否存在负环，最坏时间复杂度为 $O(nm)$ 。

常用变形技巧

例题luogu P1993 小 K 的农场

题目大意：求解差分约束系统，有 m 条约束条件，每条都为形如 $x_a - x_b \geq c_k$，$x_a - x_b \leq c_k$ 或 $x_a = x_b$ 的形式，判断该差分约束系统有没有解。

题意	转化	连边
$x_a - x_b \geq c$	$x_b - x_a \leq -c$	`add(a, b, -c);`
$x_a - x_b < c$	$x_a - x_b \leq c - 1$	`add(b, a, c-1);`
$x_a = x_b$	$x_a - x_b \leq 0,\ x_b - x_a \leq 0$	`add(b, a, 0),` `add(a, b, 0);`

跑判断负环，如果不存在负环，输出 `Yes`，否则输出 `No`。

最长路

也可以改成 $\geq$,不过要求最长路，初始值-INF

题意	转化	连边
$x_a - x_b \leq c$	$x_b - x_a \geq -c$	`add(a, b, -c);`
$x_a - x_b > c$	$x_a - x_b \geq c + 1$	`add(b, a, c+1);`
$x_a = x_b$	$x_a - x_b \leq 0,\ x_b - x_a \leq 0$	`add(b, a, 0),` `add(a, b, 0);`

二分匹配

交替路：从一个未匹配点出发，依次经过非匹配边、匹配边、非匹配边...形成的路径叫交替路。

增广路：从一个未匹配点出发，走交替路，如果途径另一个未匹配点（出发的点不算），则这条交替路称为增广路（agumenting path）。

增广路有一个重要特点：非匹配边比匹配边多一条。因此，研究增广路的意义是改进匹配。只要把增广路中的匹配边和非匹配边的身份交换即可。由于中间的匹配节点不存在其他相连的匹配边，所以这样做不会破坏匹配的性质。交换后，图中的匹配边数目比原来多了 1 条。

```
bool findp(int now)
{
    for(int i = head[now];~i;i = edge[i].next)
    {
        int to = edge[i].v;
        if(!vis[to])
        {
            vis[to] = true;
            if(matching[to] == -1 || findp(matching[to]))
            {
                matching[now] = to;
                matching[to] = now;
                return true;
            }
        }
    }
    return false;
}
```

```c
int hung(int n)
{
    int ans = 0;
    for(int i = 1;i <= n ;i++)
    {
        if(matching[i] == -1){
            memset(vis,0,sizeof(vis));
            if(findp(i))ans++;
        }
    }
    return ans;
}
```

2-SAT

> SAT 是适定性（Satisfiability）问题的简称。一般形式为 k - 适定性问题，简称 k-SAT。而当 $k > 2$ 时该问题为 NP 完全的。所以我们之研究 $k = 2$ 的情况。

定义

2-SAT，简单的说就是给出 n 个集合，每个集合有两个元素，已知若干个 $<a,b>$ ，表示 a 与 b 矛盾（其中 a 与 b 属于不同的集合）。然后从每个集合选择一个元素，判断能否一共选 n 个两两不矛盾的元素。显然可能有多种选择方案，一般题中只需要求出一种即可。

现实意义

比如邀请人来吃喜酒，夫妻二人必须去一个，然而某些人之间有矛盾（比如 A 先生与 B 女士有矛盾，C 女士不想和 D 先生在一起），那么我们要确定能否避免来人之间没有矛盾，有时需要方案。这是一类生活中常见的问题。

使用布尔方程表示上述问题。设 a 表示 A 先生去参加，那么 B 女士就不能参加（ $\neg a$ ）； b 表示 C 女士参加，那么 $\neg b$ 也一定成立（D 先生不参加）。总结一下，即 $(a \vee b)$ （变量 a, b 至少满足一个）。对这些变量关系建有向图，则有： $\neg a \Rightarrow b \wedge \neg b \Rightarrow a$ （ a 不成立则 b 一定成立；同理， b 不成立则 a 一定成立）。建图之后，我们就可以使用缩点算法来求解 2-SAT 问题了。

常用解决方法

TarjanSCC 缩点

算法考究在建图这点，我们举个例子来讲：

假设有 $a1, a2$ 和 $b1, b2$ 两对，已知 $a1$ 和 $b2$ 间有矛盾，于是为了方案自洽，由于两者中必须选一个，所以我们就要拉两条条有向边 $(a1, b1)$ 和 $(b2, a2)$ 表示选了 $a1$ 则必须选 $b1$ ，选了 $b2$ 则必须选 $a2$ 才能够自洽。

然后通过这样子建边我们跑一遍 Tarjan SCC 判断是否有一个集合中的两个元素在同一个 SCC 中，若有则输出不可能，否则输出方案。构造方案只需要把几个不矛盾的 SCC 拼起来就好了。

输出方案时可以通过变量在图中的拓扑序确定该变量的取值。如果变量 $\neg x$ 的拓扑序在 x 之后，那么取 x 值为真。应用到 Tarjan 算法的缩点，即 x 所在 SCC 编号在 $\neg x$ 之前时，取 x 为真。因为 Tarjan 算法求强连通分量时使用了栈，所以 Tarjan 求得的 SCC 编号相当于反拓扑序。

显然地，时间复杂度为 $O(n+m)$ 。

爆搜

就是沿着图上一条路径，如果一个点被选择了，那么这条路径以后的点都将被选择，那么，出现不可行的情况就是，存在一个集合中两者都被选择了。

那么，我们只需要枚举一下就可以了，数据不大，答案总是可以出来的。

爆搜模板

下方代码来自刘汝佳的白书：

```cpp
// 来源：白书第 323 页
struct Twosat {
  int n;
  vector<int> g[maxn * 2];
  bool mark[maxn * 2];
  int s[maxn * 2], c;
  bool dfs(int x) {
    if (mark[x ^ 1]) return false;
    if (mark[x]) return true;
    mark[x] = true;
    s[c++] = x;
    for (int i = 0; i < (int)g[x].size(); i++)
      if (!dfs(g[x][i])) return false;
    return true;
  }
  void init(int n) {
    this->n = n;
    for (int i = 0; i < n * 2; i++) g[i].clear();
    memset(mark, 0, sizeof(mark));
  }
```

```cpp
  void add_clause(int x, int y) {  // 这个函数随题意变化
    g[x].push_back(y ^ 1);         // 选了 x 就必须选 y^1
    g[y].push_back(x ^ 1);
  }
  bool solve() {
    for (int i = 0; i < n * 2; i += 2)
      if (!mark[i] && !mark[i + 1]) {
        c = 0;
        if (!dfs(i)) {
          while (c > 0) mark[s[--c]] = false;
          if (!dfs(i + 1)) return false;
        }
      }
    return true;
  }
};
```

例题

HDU3062Party

> 题面：有 n 对夫妻被邀请参加一个聚会，因为场地的问题，每对夫妻中只有 1 人可以列席。在 $2n$ 个人中，某些人之间有着很大的矛盾（当然夫妻之间是没有矛盾的），有矛盾的 2 个人是不会同时出现在聚会上的。有没有可能会有 n 个人同时列席？

这是一道多校题，裸的 2-SAT 判断是否有方案，按照我们上面的分析，如果 $a1$ 中的丈夫和 $a2$ 中的妻子不合，我们就把 $a1$ 中的丈夫和 $a2$ 中的丈夫连边，把 $a2$ 中的妻子和 $a1$ 中的妻子连边，然后缩点染色判断即可。

End…?

给一个数k，问他的正整数倍数中，（十进制下）每一位的和最小是多少

$$2 \leq k \leq 10^5$$

从1开始， 建立一个数x到x+1和x*10分别为1和0的边，最后找到最快到达的k的倍数，即答案。
最短路

给出一个只包含‘.’和‘*’的矩阵，用任意长度的宽为1的木板覆盖所有的‘*’而不覆盖‘.’，木板必须跟矩形的长或宽平行。问最少需要多少块木板。

行列建图

最小点覆盖 = 选取尽可能少的点，使得，所有边都和这些点集里某些边关联

最小点覆盖数 = 最大匹配数

匈牙利

给定n个区间，每个区间(ai,bi)，以及权值wi。选出一些区间，满足权值和最大且任何一个点不会被超过k个区间覆盖

区间k覆盖， 建图， 费用流

对于某个区间(a,b)，建边a->b，容量为1，费用为此区间的价值，这样意味着每个区间只能被选一次。另外对于离散化后的端点，从源点向第一个点连边，容量为k费用为0，从最后一个点向汇点连边，容量为k费用为0，然后相邻的两点i和i+1，连边i->i+1，容量大于等于k费用为0，这样建图的话，可以发现对于一个流量为1的流，选取的区间一定不会发生重叠，于是最多选取k次，满足题意要求。最后直接输出费用流结果即可

给一个n，要求在1到n中选取若干的数，且两两互质，要求和最大

选的数最多两个质因子组成

将大于$\sqrt{n}$的质数和小于$\sqrt{n}$的质数建边，跑费用流

~~图论教三大教义（并不）~~

1. ~~任何NP问题都可以转化成判断图中是否存在哈密顿回路~~
2. ~~任何DP都可以转化成DAG上求最短路~~
3. ~~任何问题都能转化成网络流~~

END