限制条件的图论模型转化

黄嘉盛

2022 年 7 月 13 日

问题简述

差分约束系统是特殊的 n 元一次不等式组，不等式组的每一个不等式称为一个约束条件。

通过不等式的变形，可以通过最短路算法对差分约束系统进行求解。

问题简述

差分约束系统是特殊的 n 元一次不等式组，不等式组的每一个不等式称为一个约束条件。

通过不等式的变形，可以通过最短路算法对差分约束系统进行求解。

常见的差分约束，形式为 $a_i - a_j \leq p_k$ ，其中 a_i, a_j 为 i, j 的值，未被确定，p_k 为一开始就给定的值，你需要判断的是是否存在分配每个 a_i 的方案，使得每一个限制都被满足。

模型转化

观察差分约束的常见形式 $a_i - a_j \leq p_k$ ，并将其变形为

$a_i \leq p_k + a_j$

模型转化

观察差分约束的常见形式 $a_i - a_j \leq p_k$ ，并将其变形为

$a_i \leq p_k + a_j$

此时联想到最短路中需要满足的不等式（设 d_x 为到 x 的最短路长度，$dist(u, v)$ 为 u, v 之间距离），则有 $d_y \leq d_x + dist(x, y)$ ，与差分约束不等式的变形是形式一致的。

模型转化

观察差分约束的常见形式 $a_i - a_j \leq p_k$，并将其变形为

$a_i \leq p_k + a_j$

此时联想到最短路中需要满足的不等式（设 d_x 为到 x 的最短路长度，$dist(u, v)$ 为 u, v 之间距离），则有 $d_y \leq d_x + dist(x, y)$，与差分约束不等式的变形是形式一致的。

所以可以新建 n 个点，对于限制 $a_i - a_j \leq p_k$，在图中由 j 向 i 连一条边，距离为 p_k，此时约束问题变成了图上的最短路问题。

其他形式

差分约束还有可能有其他的形式，比如 $a_i - a_j \geq p_k$ 。

其他形式

差分约束还有可能有其他的形式，比如 $a_i - a_j \geq p_k$ 。

运用相同的变形，变为 $a_i \geq a_j + p_k$ 则转化为最长路问题，或者变形为 $a_j \leq a_i + (-p_k)$ ，则仍为最短路问题。

判断解的存在性

通过上面两种形式可以将约束问题转化为最短路问题，但是显然 p_k（或 $-p_k$）可能是负数，那么最短路不使用 dijkstra 算法，而使用 SPFA 算法

判断解的存在性

通过上面两种形式可以将约束问题转化为最短路问题，但是显然 p_k（或 $-p_k$）可能是负数，那么最短路不使用 dijkstra 算法，而使用 SPFA 算法

存在负权边也就说明可能存在负环，在这种情况下问题肯定是无解的。

判断解的存在性

通过上面两种形式可以将约束问题转化为最短路问题，但是显然 p_k （或 $-p_k$）可能是负数，那么最短路不使用 dijkstra 算法，而使用 SPFA 算法

存在负权边也就说明可能存在负环，在这种情况下问题肯定是无解的。

具体判断方法只需要在 SPFA 的 BFS 时对每个点记录一个 cnt 数组，在更新 $d_y = d_x + dist(x, y)$ 时也更新 $cnt_y = cnt_x + 1$ ，当 $cnt_i > n$ 时说明一定出现了负环，也说明约束问题无解

求具体解

显然，对于差分约束系统的一组解 $\{a_i|i\in[1,n]\},\{a_i+\Delta|i\in[1,n]\}$ 也是一组解（因为 Δ 会在作差时被消掉）。

因此，为了判断差分约束系统是否有解，我们可以先求一组负数解，即增加一个编号为 0 的节点，令 $a_0=0$，并从 0 号节点向每个节点连一条边，这样就可以保证 $\forall i,a_i\le 0$。

求具体解

显然，对于差分约束系统的一组解 $\{a_i|i\in[1,n]\},\{a_i+\Delta|i\in[1,n]\}$ 也是一组解（因为 Δ 会在作差时被消掉）。

因此，为了判断差分约束系统是否有解，我们可以先求一组负数解，即增加一个编号为 0 的节点，令 $a_0=0$，并从 0 号节点向每个节点连一条边，这样就可以保证 $\forall i,a_i\le 0$。

以 0 号节点为起点跑最短路，显然，若图中存在负环，则说明永远满足不了所有的约束条件，差分约束系统无解；否则 $\{a_i|i\in[1,n]\}$ 就是一组解。

luogu P1993 小 K 的农场

小 K 在 MC 里面建立很多很多的农场，总共 n 个，以至于他自己都忘记了每个农场中种植作物的具体数量了，他只记得一些含糊的信息（共 m 个），以下列三种形式描述：

- 农场 a 比农场 b 至少多种植了 c 个单位的作物；
- 农场 a 比农场 b 至多多种植了 c 个单位的作物；
- 农场 a 与农场 b 种植的作物数一样多。

但是，由于小 K 的记忆有些偏差，所以他想要知道存不存在一种情况，使得农场的种植作物数量与他记忆中的所有信息吻合。

$1 \leq n, m, a, b, c \leq 5 \times 10^3$

luogu P1993 小 K 的农场 | Solution

对于 $x_a - x_b \leq c$ 和 $x_a - x_b \geq c$ 都按照前面的方法转化就行了，

对于 $x_a = x_b$ 就直接变为 $x_a - x_b \leq 0$ 以及 $x_a - x_b \geq 0$

然后就是板子

Teleport

有 $n(n \leq 10^6)$ 个城市，城市间互相连接形成了一棵树，每条树边有边权 w_i 为走过这条边需要花费的时间。

每个点有传送装置，并且 i 点传送装置有一个参数 a_i ，从 i 传送到 j 需要花费时间 $|a_i - a_j|$ 。

管理者不希望两点之间传送花费的时间比走路时间还多，而且由于城市地质限制，i 城市的参数 a_i 需要满足在 $[l_i, r_i]$ 之间。

当然你也可以对城市进行改造，用 x 的代价让所有城市接受的区间变为 $[l_i - x, r_i + x]$ ，x 非负整数

问在不进行改造的情况下，能否找到一种安排 a_i 的方案，满足上述所有要求。还需要回答在允许进行改造的情况下，最少花费多少代价进行改造，可以找到一种方案满足要求。

Teleport | Soltion

新建一个 0 号点，向 i 号点连一条长度为 r_i 的单向边，i 号点向它连一条长度为 $-l_i$ 的单向边，那么有解等价于图中不存在负环。

由于所有负边都一定连向 0 号点，图中存在负环当且仅当图中存在一个经过 0 号点的简单负环。

到这一步就可以开始二分答案 x，然后判断更新后的图中是否存在负环。

但是可以发现二分是没有必要的，因为所有经过 0 的简单环增加的长度都是 $2 \times mid$，所以可以直接找出长度最小的经过 0 号点的简单环，假设这个环长度为 l，则答案为 $max(0, \lfloor \frac{l}{2} \rfloor)$

找出长度最小的简单环可以通过 dijkstra 或树形 DP 解决，复杂度 $O(nlogn)$ 或 $O(Tn)$。

问题简述

现在有 n 个布尔变量，每个变量只能取值真或者假，所谓 k-SAT 问题就是最多存在一个问题含有 k 个布尔变量，这个问题要求这 k 个布尔变量在经过一系列位运算后满足为真或假

2-SAT 问题，就是对于每一个条件，最多只于两个变量的真假性有关，比如要求 $x \vee y = 1$ ，即要求 x 为真或 y 为真

模型转化

继续看上面 2-SAT 的这个例子，$x \vee y = 1$ 蕴含的是：

- x 为假，那么 y 肯定为真
- y 为假，那么 x 肯定为真

模型转化

继续看上面 2-SAT 的这个例子，$x \vee y = 1$ 蕴含的是：

- x 为假，那么 y 肯定为真
- y 为假，那么 x 肯定为真

于是将原图每个 x 拆成两个点，分别代表它为假和它为真。（为了方便表示，分别标号为 2x 和 2x+1。）

然后我们就可以得到 2x→2y+1 和 2y→2x+1 这两条边。不难发现此处边的意义就是“推导出”

模型转化

对于其他的约束情况也都可以这样做。如：

- x 为真时，y 为假，那么就有一条 2x+1→2y 的边；
- x 和 y 状态要一样，2x→2y 和 2x+1→2y+1 的边。

由于 2-SAT 要求能够逆向推导，所以需要保证图的对称性，利用命题与其逆否命题一定等价，可以由 $a \to b$ 得到一条新边 $\neg b \to \neg a$

2-SAT 模型求解

首先是一个很直观的暴力算法

枚举每个点的状态，真或者假（如果它之前没被标过）。然后将它能标记状态的全都标记一遍，途中如果碰到自己这个状态已经被标记就返回真，如果反状态被标记了就返回假。

如果本状态当前假设状态不行，就将它刚刚标出来的状态全部清空掉（用个栈来实现），然后继续尝试它的一个反状态，如果反状态还是不行，那就是无解。

最多尝试 n 个点，每次要最多尝试 m 条边，所以总复杂度最差是 $O(nm)$ 的。（能卡到最差，但是对于随机数据跑得比较快）

2-SAT 模型求解

如果对于一个点 x 的两个状态真或假都在一个强联通分量中，那么真假就能够互推了，肯定不成立。

然后可以用反向的拓扑序（从出度为零的开始取）来染色求解，对于一个并未染色的点将它染成 1，它的一个对立状态染成 2。然后不断重复。最后所有为 1 的就可以当做一组解了。

2-SAT 模型求解

这个成立时因为，每个条件都是两两关系，只要保证了存在解，那么这样绝对是可行的。

但没有必要写个拓扑排序，因为 Tarjan 的访问其实是根据拓扑序来的，所以最后给他的 scc 标号（sccno）正好和拓扑序相反。

如果对于一个 x，它的 sccno 比它的反状态 $x \oplus 1$ 的 sccno 要小，那么我们用 x 这个状态当做答案，否则用它的反状态当做答案。也就是我们用 sccno 小的状态当做答案。

UVALive - 3713 Astronauts

给定 n 个宇航员的年龄，平均年龄为 ave，根据下列要求分配任务：

B 任务只能分配给年龄 <ave 的宇航员；A 任务只能分配给年龄 >=ave 的宇航员；C 任务可以任意分配。

给定 m 组互相憎恨的宇航员，要求他们不能分配到同一个任务。

问能否存在这样的一组任务分配。

$1 \leq n, m \leq 10^5$

UVALive - 3713 Astronauts | Solution

每个宇航员都只能分配两种任务中的一种：A 或 C（年龄大于等于 ave），B 或 C（年龄小于 ave），那么为每个宇航员设立一个变量 x_i，x_i 为 0 表示分配 C 任务，为 1 则分配 A 或 B（根据年龄）。

对于互相仇恨的宇航员，如果属于同一类型，那么应满足 x_i 和 x_j 一真一假；如果类型不同只需要满足不同时分配 C 任务就可，即 x_i, x_j 不同时为 0。

然后建图跑 2-SAT

CF 587D Duff in Mafia

给定一张 n 个点 m 条边的无向图，每条边有一个颜色 c 和权值 t。

你要选出一些边，使得它们是一个匹配，同时剩下的边每种颜色对应的所有边也是匹配（没有两条边有同样的端点）。

同时，你要最小化选出的边的最大权值。

$n, m \leq 5 \times 10^4$。

CF 587D Duff in Mafia | Solution

首先二分答案，若此时二分的值为 m，则所有 $> m$ 的边都不能选。每条边有选和不选两种状态，所以考虑 2-SAT。设第 i 条边的状态为 x_i 表示选择它，x'_i 表示不选。连边如下：

- 对于一定不能选的边，连边 $x_i \to x'_i$ 。
- 对于选出的边一定要是一个匹配，考虑一个点 p 上的所有边 $x_{1\ldots k}$，连边 $x_i \to x'_j (i \neq j)$ 。
- 对于剩下的边每种颜色也一定要是一个匹配，考虑一个点 p 上颜色相同的所有边 $x_{1\ldots k}$，连边 $x'_i \to x_j (i \neq j)$ 。

CF 587D Duff in Mafia | Solution

但这样做第 2,3 类边的边数为 $\mathcal{O}\left(\mathrm{m}^2\right)$, 这里介绍一个 2-SAT 中比较常用的优化——前缀优化。

先考虑第 2 类边, 设 $x_{1\ldots k}$ 的前缀点为 $s_{1\ldots k}$, $x'_{1\ldots k}$ 的后缀点为 $s'_{1\ldots k}$。连边如下:

- $x_i \to s_i, s'_i \to x'_i$ 。
- $s_{i-1} \to s_i, s'_i \to s'_{i-1}$ 。
- $s_{i-1} \to x'_i, x_i \to s'_{i-1}$ 。

这样建图和第 2 类边是等效的。而第 3 类边，就把第 2 类边的箭头全部反过来即可。

然后就是 2-SAT 了

最初想法

如果枚举每个起点用 Bellman-Ford 跑单源最短路，时间复杂度 $O(n^2m)$，直接使用 Floyd 算法，时间复杂度 $O(n^3)$

注意到单源最短路中 Dijkstra 的时间复杂度表现优秀，尝试用枚举每个源点跑 Dijkstra 的方法，发现无法处理负环

于是 Johnson 想出了一个方法，给每条边重新赋权 (reweight)，使得边权均为非负数，并且原图中的最短路在新图中仍为最短路

重新赋权

对与重新赋权使得边权均非负，最简单的想法是给每条边都加上权值 w，但是这样显然是错的，如对与一条权值和为 w_1 长度为 l_1 的路径和 (w_2, l_2) 的路径，都加上权值 w 后两条路径权值和分别变为了 $w_1 + w \times l_1$ 和 $w_2 + w \times l_2$，可能导致大小顺序颠倒

Johnson 算法则通过另外一种方法来给每条边重新标注边权：

- 新建源点 S，并且由源点向每个点连长度为 0 的边
- S 为源点跑 Bellman-Ford，得到 S 到每个节点的最短路长度 h_u
- 对于每条边 (u, v, w)，将边权重赋值为 $w + h_u - h_v$

重新赋权后能保证权值均为非负（正确性见网上势能分析），然后枚举每个点作为源点跑 Dijkstra 就可以了，复杂度 $O(nm \log m)$