图论二

单源最短路及 Floyd-Warshall 算法

郑欣

2024 年 7 月 26 日

最短路

- 带权图：$G=(V,E,w)$，其中 $w:E\to\mathbb{R}$ 为边权。
- 单源最短路：给定起点 $s\in V$，对于所有 $t\in V$，求出 s 到 t 的最短路 $\mathrm{dis}(s,t)$。
  - Bellman-Ford (SPFA)
  - Dijkstra
- 全源最短路：对于所有 $s,t\in V$，求出 s 到 t 的最短路 $\mathrm{dis}(s,t)$。
  - Floyd-Warshall
  - Johnson

1 全源最短路

- Floyd

2 单源最短路

3 例题

Floyd 算法

用 $f_{k,i,j}$ 表示 i 到 j 仅经过前 k 个点的最短路（i 和 j 可以不在前 k 个点中）。

- 初始状态：
  - 对所有 $i \in V$：$f_{0,i,i} = 0$；
  - 对所有 $(i,j) \in E$：$f_{0,i,j} = w(i,j)$；
  - 对所有 $(i,j) \notin E$ 且 $i \neq j$：$f_{0,i,j} = +\infty$；
- 转移：$f_{k,i,j} = \min\{f_{k-1,i,j}, f_{k-1,i,k} + f_{k-1,k,j}\}$。

可以求出所有点对之间的最短路，边权任意（无负环）。复杂度 $O(n^3)$。

Code

```
void floyd() {
    // f[i][j]: i 到 j 的最短路，初始化为边权或 0 (若 i = j) 或 +∞ (如果没有这条边)
    for (int k = 1; k <= n; ++k) // 一定要先枚举 k
        for (int i = 1; i <= n; ++i)
            for (int j = 1; j <= n; ++j)
                f[i][j] = min(f[i][j], f[i][k] + f[k][j]);
}
```

例题 1

Luogu B3611 传递闭包

给一个有向图，对于任意两点 u, v 问 u 是否能到达 v。

范围：$n \le 100, m \le n^2$

用 $f_{i,j} = 1$ 表示 i 能够到达 j。类似 Floyd 算法计算若仅经过前 k 个点，i 能否到 j：

- 初始状态：若 $(i,j) \in E$，则 $f_{0,i,j} = 1$，否则 $f_{0,i,j} = 0$。
- 转移：$f_{k,i,j} = f_{k-1,i,j} \vee (f_{k-1,i,k} \wedge f_{k-1,k,j})$。

Code (复杂度 $O(n^3)$)

```
void floyd() {
    for (int k = 1; k <= n; ++k)
        for (int i = 1; i <= n; ++i)
            for (int j = 1; j <= n; ++j)
                f[i][j] |= f[i][k] & f[k][j];
}
```

例题 1

Code (复杂度 $O(n^3)$)

```
void floyd() {
    for (int k = 1; k <= n; ++k)
        for (int i = 1; i <= n; ++i)
            if (f[i][k])
                for (int j = 1; j <= n; ++j)
                    f[i][j] |= f[k][j];
            /*
             * for (int j = 1; j <= n; ++j)
             *     f[i][j] |= f[i][k] & f[k][j];
             */
}
```

用 bitset 压缩 01 数组可以一次性计算 $w = 64$ 位。

Code (Bitset 优化，复杂度 $O(n^3/w)$)

```
void floyd() {
    // bitset<N> f[N];
    for (int k = 1; k <= n; ++k)
        for (int i = 1; i <= n; ++i)
            if (f[i][k]) f[i] |= f[k];
}
```

例题 2

Luogu P6464　传送门

给定一个无向带权图，添加一条长度为 0 的无向边，使得所有点对距离之和最小。

范围：$n \leq 100, m \leq \binom{n}{2}$

首先 Floyd 求出原图的点两两之间的最短路 dis。

假设添加边 (u, v)，则 i 到 j 的最短路变为

$$
\begin{aligned}
&\min\{\mathrm{dis}(i,j), && \text{直接从 } i \text{ 到 } j \\
&\quad \mathrm{dis}(i,u)+\mathrm{dis}(v,j), && \text{先从 } i \text{ 到 } u\text{，然后从 } v \text{ 到 } j \\
&\quad \mathrm{dis}(i,v)+\mathrm{dis}(u,j)\} && \text{先从 } i \text{ 到 } v\text{，然后从 } u \text{ 到 } j
\end{aligned}
$$

枚举 (u, v)，$O(1)$ 算出每对 (i,j) 的最短路，总时间复杂度 $O(n^4)$。

例题 3

Luogu P1119　灾后重建

给定一个无向带权图，每个点有一个时间戳 t_i。有 Q 组询问，每组询问包含两个点 $x, y \in V$ 和时间 t，询问从 x 到 y 仅经过时间戳不超过 t 的点的最短路。

范围：$n \leq 200, m \leq \binom{n}{2}, Q \leq 5 \cdot 10^4$

只能经过 $t_i \leq t$ 的点，可以联想到 Floyd。

按 t_i 从小到大加入点 i 跑 Floyd，此时 dis 数组的含义即为仅经过 $\leq t_i$ 的点时的最短路。

询问 t 时加入所有 $t_i \leq t$ 的点即可。需要注意 $t_x > t$ 或 $t_y > t$ 时输出 -1。

Luogu P6175　无向图的最小环问题

给定一个无向带权图，求边权之和最小的简单环（即一个不能经过重复的点的环）。

范围：$n \le 100, m \le \binom{n}{2}$。

暴力：枚举删除哪条边 (u, v)，求 u 到 v 的最短路。复杂度 $O(m^2 \log m)$。

考虑怎么让最短路不经过某个点/边。

枚举环上编号最大的点，假设编号为 k，则 Floyd 算法在第 $(k-1)$ 轮求出的最短路一定不会经过 k。因此此时 $u \rightsquigarrow v \to k \to u$ 一定是简单环。枚举 (u, v)，求 $f_{k,u,v} + w(v, k) + w(k, u)$ 的最小值即可。复杂度 $O(n^3)$。

有向图的最小简单环：$\min_{u \ne v} \mathrm{dis}_{u,v} + \mathrm{dis}_{v,u}$

1 全源最短路

2 单源最短路

- Bellman-Ford
- Dijkstra

3 例题

观察：如果没有负环，则任意两点之间的最短路经过至多 $(n-1)$ 条边。

假设起点为 s。用 $f_{k,u}$ 表示 s 到 u 经过至多 k 条边的最短路。

- 初始状态：$f_{0,s}=0$，$f_{0,u}=+\infty\ (s\neq u)$。
- 转移（松弛操作）：$f_{k,v}=\min\{f_{k-1,v},\min_u(f_{k-1,u}+w(u,v))\}$。

可以求出给定起点到所有点的最短路，边权任意（无负环）。复杂度 $O(nm)$。

Code

```
void bellman(int s) {
    // G[u]: u 的邻居及边权, vector<pair<int, int>> 类型
    vector<int> dis(n + 1, INF);                     // dis[u]: 从 s 到 u 的最短路
    dis[s] = 0;
    for (int k = 1; k < n; ++k) {                    // 进行 (n − 1) 轮松弛操作
        for (int u = 1; u <= n; ++u) {               // 枚举所有目前可以到达的点
            if (dis[u] >= INF) continue;
            for (auto [v, w]: G[u]) {                // u 到 v 有一条长度为 w 的有向边
                dis[v] = min(dis[v], dis[u] + w);    // 松弛
            }
        }
    }
}
```

在 Bellman-Ford 中，只有在上一轮中距离被更新的节点，才有可能导致下一轮距离的更新。

SPFA (Shortest Path Faster Algorithm)：用队列维护哪些节点将会参与松弛操作。

很多时候 SPFA 运行效率高于朴素的 Bellman-Ford，但最坏情况下复杂度还是 $O(nm)$。

Code

```
void SPFA(int s) {
    vector<int> dis(n + 1, INF);
    queue<int> q;
    vector<int> vis(n + 1);  // vis[u]: u 是否在队列中，避免同一轮松弛中同一个节点多次进入队列
    dis[s] = 0;
    q.push(s), vis[s] = 1;
    while (!q.empty()) {
        int u = q.front();
        q.pop(), vis[u] = 0;
        if (dis[u] >= INF) continue;
        for (auto [v, w]: G[u]) {
            if (dis[v] <= dis[u] + w) continue;
            dis[v] = dis[u] + w;
            if (!vis[v])
                q.push(v), vis[v] = 1;  // v 的距离被更新且不在队列中，因此加入队列
        }
    }
}
```

例题 5

Luogu P3385　负环

给定一个有向图，问是否存在能从 1 号节点到达的负环（边权之和 < 0 的环）。

范围：$n \le 2 \cdot 10^3, m \le 3 \cdot 10^3$

当没有负环时最短路长度不超过 $n-1$，因此每个节点的距离更新次数不超过 $n-1$，因此入队次数不超过 $n-1$。

如果从 1 号节点开始跑 SPFA，发现某个节点入队次数 $\ge n$，说明存在 1 号节点可达的负环。时间复杂度 $O(nm)$。

如果要判断整个图中有没有负环：增加一个超级源点 s 连接所有点（$1 \sim n$），以 s 为源点跑 SPFA。

例题 5

Code (负环)

```
bool SPFA(int s) {
    vector<ll> dis(n + 1, INF);
    queue<int> q;
    vector<int> vis(n + 1);
    vector<int> cnt(n + 1);   // cnt[u]: 记录点 u 的入队次数
    dis[s] = 0;
    q.push(s), vis[s] = 1, cnt[s] = 1;
    while (!q.empty()) {
        int u = q.front();
        q.pop(), vis[u] = 0;
        if (dis[u] >= INF) continue;
        for (auto [v, w]: G[u]) {
            if (dis[v] <= dis[u] + w) continue;
            dis[v] = dis[u] + w;
            if (!vis[v]) {
                q.push(v), vis[v] = 1;
                ++cnt[v];                      // 记录 v 的入队次数
                if (cnt[v] > n) return false;  // 点 v 入队超过 n 次说明有负环
            }
        }
    }
    return true;
}
```

例题 6

HNOI2009　最小圈

给一个带权有向图，求平均边权最小的环。

范围：$n \le 3 \cdot 10^3, m \le 10^4$

考虑二分答案。假设答案为 w，则存在一个平均权值 $< w$ 的环当且仅当把所有边权减 w 后存在负环。因此二分 w 然后用 SPFA 判负环即可。复杂度 $O(nm \log \varepsilon^{-1})$。

事实上有以下结论：

$$\text{ans} = \min_{v \in V} \max_{0 \le k < n} \frac{f_{n,s,v} - f_{k,s,v}}{n-k},$$

其中 s 为超源。直观理解就是长为 k 的最短路在最坏情况下会比长为 n 的最短路少走若干个环。因此 Bellman-Ford 对每个 k 求出长为 k 的最短路即可。复杂度 $O(nm)$。

例题 7

Luogu P2886　Cow Relays

给定一个无向连通图，求起点 s 到终点 t 经过恰好 k 条边的最短路。

范围：$n, m \leq 100, k \leq 10^6$

用 $f_{k,u}$ 表示 s 到 u 经过恰好 k 条边的最短路。

状态转移：$f_{k,v} = \min_u(f_{k-1,u} + w(u,v))$。

时间复杂度 $O(mk)$，可以用滚动数组将空间复杂度降低至 $O(m)$。

例题 7

矩阵快速幂做法：

用 $f_{k,u,v}$ 表示从 u 到 v 恰好经过 k 条边的最短路。

- 初始状态：若 $(u,v)\in E$ 则 $f_{1,u,v}=w(u,v)$，否则 $f_{1,u,v}=+\infty$。
- 转移：$f_{k+\ell,u,v}=\min_w(f_{k,u,w}+f_{\ell,w,v})$（长为 k 和 ℓ 的路径拼接成长 $k+\ell$ 的路径）。

把 f_k 看作一个 $n\times n$ 的矩阵，则该转移式可以看作 $(\min,+)$ 的矩阵乘法。用矩阵快速幂优化可以在 $O(n^3\log k)$ 时间内得到 f_k：

- $f_{2k,u,v}=\min_w(f_{k,u,w}+f_{k,w,v})$。
- $f_{2k+1,u,v}=\min_w(f_{2k,u,w}+f_{1,w,v})$。

维护当前能够确定最短路的点集 S:

- 初始状态：$S = \varnothing$，$\mathrm{dis}_s = 0$。
- 每次找到一个不在 S 中且 dis_u 最小的点 u 加入 S，并对每个 $(u, v) \in E$ 进行松弛操作

$$\mathrm{dis}_v \leftarrow \min\{\mathrm{dis}_v, \mathrm{dis}_u + w(u, v)\},$$

直到 $S = V$ 时即可得到所有点的 dis。

可以求出给定起点到所有点的最短路，边权为正。

正确性证明

我们用归纳法说明所有 S 中的点 $u \in S$ 满足 dis_u 是 s 到 u 的最短路：

- 当 $S = \varnothing$ 时，结论显然成立。
- 当 $S \neq \varnothing$ 时，假设 v 是不在 S 中 dis_v 最小的点。从 s 到 v 的最短路不可能经过 S 外的点，因为边权为正，且 dis_v 在 $v \notin S$ 中最小。因此 s 到 v 的最短路上的点除了 v 全在 S 内。假设 $s \rightsquigarrow u \to v$，则 $\mathrm{dis}_v = \min_{u \in S} \mathrm{dis}_u + w(u, v)$。因此可以将 v 加入 S，此时 dis_v 也是最短路。

代码实现

Code (朴素实现，复杂度 $O(n^2)$)

```
void dijkstra(int s) {
    vector<int> dis(n + 1, INF);  // dis[u]: 从 s 到 u 的最短路
    vector<int> vis(n + 1);       // vis[u]: u 是否在集合 S 中
    dis[s] = 0;
    for (int T = 1; T <= n; T++) {
        int u = 0, mind = INF;
        // 找到不在 S 中 dis 最小的点 u
        for (int i = 1; i <= n; i++) {
            if (!vis[i] && dis[i] < mind) u = i, mind = dis[i];
        }
        vis[u] = 1;                   // 把点 u 加入 S
        for (auto [v, w]: G[u]) {
            if (dis[v] > dis[u] + w) {
                dis[v] = dis[u] + w;
            }
        }
    }
}
```

代码实现

用小根堆维护所有被更新过距离的点。

每条边会被访问一次且导致最多一个点插入小根堆，总复杂度 $O(m\log m)$。

Code (堆优化 Dijkstra，复杂度 $O(m\log m)$)

```
void dijkstra(int s) {
    vector<int> dis(n + 1, INF);  // dis[u]: 从 s 到 u 的最短路
    vector<int> vis(n + 1);       // vis[u]: u 是否在集合 S 中
    // q: 小根堆，pair<int, int> 存放 (距离, 点)
    priority_queue<pair<int, int>, vector<pair<int, int>>, greater<>> q;
    dis[s] = 0;
    q.emplace(0, s);
    while (!q.empty()) {
        auto [_, u] = q.top();    // 取出当前距离最小的点 u
        q.pop();
        if (vis[u]) continue;     // 保证点 u 不在 S 里面
        vis[u] = 1;               // 把点 u 加入 S
        for (auto [v, w]: G[u]) {
            if (dis[v] > dis[u] + w) {
                dis[v] = dis[u] + w;
                q.emplace(dis[v], v);
            }
        }
    }
}
```

01 BFS

BFS：边权为 1 求最短路。

01 BFS：边权为 0 或 1 求最短路。

用 deque 或两个 queue 替代 Dijkstra 算法中的优先队列。边权为 0 加入队首，否则加入队尾。复杂度 $O(m)$。

对比

三种算法的比较：

- Floyd：全源最短路，任意边权，时间复杂度 $O(n^3)$。
- Bellman-Ford (SPFA)：单源最短路，任意边权，时间复杂度 $O(nm)$。
- Dijkstra：单源最短路，边权非负，时间复杂度 $O(m \log n)$。

1 全源最短路

2 单源最短路

3 例题

- 基础建图
- 建图优化
- 最短路树
- 差分约束

例题 1

Luogu P1629　邮递员送信

给一个有向正权图，对每个 $2 \le i \le n$，求从 1 到 i 然后返回 1 的最短路长度。

范围：$n \le 10^3, m \le 10^5$

从 1 号点开始跑 Dijkstra 可以求出所有 $\mathrm{dis}(1, i)$。

原图的 $\mathrm{dis}(i, 1)$ 相当于反图（即把所有边反向建图）的 $\mathrm{dis}(1, i)$，因此在反图上从 1 号点跑 Dijkstra 即可求出所有 $\mathrm{dis}(i, 1)$。

复杂度 $O(m \log m)$。

例题 2

NOIP2009　最优贸易

给一个有向图，每个点有一个点权 w_i。你需要从 1 出发，最后到 n，途中可以进行以下操作至多一次：选择一个点 a 以价格 w_a 买入水晶球，然后在选择一个点 b 以加个 w_b 卖出。求最大利润。

即选择两个点 a, b，使得存在 $1 \rightsquigarrow a \rightsquigarrow b \rightsquigarrow n$ 的路径，且 $w_b - w_a$ 最大。

范围：$n \le 10^5, m \le 5 \cdot 10^5$

把原图复制 3 份表示 3 种状态：还没买入、已经买入但没卖出、已经卖出。记点 v 在三层中对应的点分别为 v_1, v_2, v_3：

- 在 v 处买入水晶球：$v_1 \to v_2$，边权 $-w_v$；
- 在 v 处卖出水晶球：$v_2 \to v_3$，边权 w_v；

则 1_1 到 n_3 的最长路则表示最大总利润。

例题 3

JLOI2011　飞行路线

给一个无向正权图，你可以把至多 k 条边的边权变成 0，求 s 到 t 的最短路。

范围：$n \le 10^4, m \le 5 \cdot 10^4, k \le 10$

把原图复制 $(k+1)$ 份，其中第 i $(0 \le i \le k)$ 层表示已经将 i 条边的边权变成 0。记原图中的点 v 在第 i 层中的复制为 v_i。

对于原图中的每条边 (u, v)，连接 (u_i, v_{i+1})，边权为 0，表示耗费一次机会免费经过 (u, v)。

从 s_0 出发跑最短路，则答案为 $\min_{0 \le i \le k} \operatorname{dis}(s_0, t_i)$。

NOIP2017　逛公园

给一个有向带权图，边权非负。求 1 到 n 的长度不超过 $d+k$ 的路径数量 $\bmod P$，其中 d 是 1 到 n 的最短路长度。如果有无穷多条合法路径输出 -1。

范围：$n \le 10^5, m \le 2 \cdot 10^5, k \le 50$

首先从 1 开始跑 Dijkstra 得到 1 到 i 的最短路 dis_i。

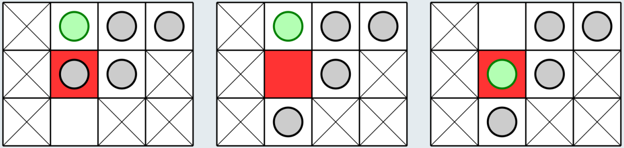
将原图的节点复制 $\mathrm{dis}_n + k$ 份构造分层图，第 i 层表示路径长度为 i。对于原图中的每条权值为 w 的边 (u, v)，对所有 i 连接有向边 (u_i, v_{i+w})。则答案为从 1_0 到所有 n_i $(\mathrm{dis}_n \le i \le \mathrm{dis}_n + k)$ 的路径数量之和，如果有环答案为 -1。

直接这样做复杂度是 $O(m(\mathrm{dis}_n + k))$。注意到只有当 $\mathrm{dis}_u \le i \le \mathrm{dis}_u + k$ 时点 u_i 是有用的，因此可以只保留每个点的 k 个复制。复杂度 $O(mk)$。

例题 5

NOIP2013　华容道

给一个 $n \times m$ 的棋盘，每个格子可能是障碍物或棋子或空的，且有且仅有一个空格。现在有 q 组询问，每组询问给定空格、目标棋子（绿色）、目标位置（红色）的坐标，你需要回答将目标棋子移动到目标位置所需的最小步数。



初始状态　　第一步之后　　第二步之后

范围：$n, m \leq 30, q \leq 500$

例题 5

观察 1：移动目标棋子必须空格在棋子旁边，且只能往空格所在方向移动。

观察 2：空格只要不经过目标棋子，可以上下左右任意移动，且不改变目标棋子的位置。

预处理：枚举目标棋子所在的位置，预处理空格从一个方向移动到另一个方向的最短路。预处理复杂度 $O(16(nm)^2)$。

对每组询问建图跑最短路，将 (棋子坐标, 空格方向) 作为顶点，建两种边：

- 目标棋子不动，改变空格方向，边权在预处理时已经求出。
- 目标棋子向空格方向移动，移动后空格反向，边权为 1。

注意空格一开始可能不在棋子旁边，所以需要建一个超源连接目标棋子起点的四个方向，边权为将空格移动到棋子旁边的代价。

每组询问复杂度 $O(nm\log nm)$，总复杂度 $O((nm)^2+qnm\log nm))$。

Luogu P4366 最短路

给一个 n 个顶点的图，顶点编号为 1 到 n。对于所有 $1 \leq i,j \leq n$，i 到 j 有一条权值 $c \cdot (i \oplus j)$ 的边（$\oplus$ 表示异或）。除此之外，图中还有 m 条有向边，权值给定。求 a 到 b 的最短路。

范围：$n \leq 10^5, m \leq 5 \cdot 10^5$

直接建图会有 $O(n^2 + m)$ 条边，复杂度太高。

不是所有边都是有必要的：对于某条边 (x,y)，如果不经过 (x,y) 仍有 $\mathrm{dis}(x,y) \leq w(x,y)$，那么可以删除 (x,y) 这条边。

例如 $w(2,7) = 5$，而事实上 $2 \to 3 \to 7$ 的总长度也只有 5，因此可以删除 $(2,7)$。

由于异或操作中每个二进制位是独立的，因此可以每次只改变一个二进制位，达到与一次异或相同的效果。即对于每个 x，只保留 $(x, x \oplus 2^i)$ $(i \leq 18)$。此时图中只有 $O(n \log n + m)$ 条边。

例题 7

线段树优化最短路

给一个 n 个顶点的图，进行 m 次操作，每次操作将 $[\ell_i, r_i]$ 内的所有点向 $[u_i, v_i]$ 内所有点连权值 w_i 的有向边。求源点 s 到其它点的最短路。

范围：$n \le 10^5, m \le 10^5$

建两棵线段树：

- 第一棵线段树 T_{in} 从根向叶子节点连边，边权为 0。树上的一个节点表示对应区间能访问所有叶子节点；
- 第二棵线段树 T_{out} 从叶子节点向根连边。节点表示所有在对应区间中的叶子节点都能访问这个区间。

对于 $[\ell, r] \to [u, v]$，在 T_{out} 上分解 $[\ell, r]$，然后全部连到一个虚点；在 T_{in} 上分解 $[u, v]$，然后从另一个虚点连到 $[u, v]$ 的分解。将两个虚点之间连边权 w 的边。这样每次操作仅新增 $O(\log n)$ 条边。

模板题：CF786B Legacy

例题 8

CF 1076D　Edge Deletion

给定一个无向带权图，要求仅保留 k 条边，使得从 1 到尽可能多的点最短路保持不变。

范围：$n, m \leq 3 \cdot 10^5, k \leq m$

最短路树

最短路树：一棵 s 为根的（外向）生成树，满足 s 到所有点都存在一个只经过树边的最短路。

求 s 为起点的最短路 dis，构造原图的一个子图 G'：若 $\mathrm{dis}_v = \mathrm{dis}_u + w(u, v)$，则连一条 $u \to v$ 的边。这个子图有以下性质：

- G' 是一个 DAG。
- G' 上从 s 出发的任意一条路都是最短路。

因此 G' 的任意生成树都是最短路树。

CF 1076D　Edge Deletion

给定一个无向带权图，要求仅保留 k 条边，使得从 1 到尽可能多的点最短路保持不变。

范围：$n, m \le 3 \cdot 10^5, k \le m$

求出 1 为根最短路树，尽可能删非树边，非树边删完删叶子节点，这样有 $\min\{k+1, n\}$ 个点的最短路不变。

计数

最短路树计数：每个点选一个入度可以构成一棵最短路树，因此为 DAG 上每个点的入度乘积。令 $\deg_v$ 为满足 $\mathrm{dis}_u + w(u,v) = \mathrm{dis}_v$ 的点 u 的数量，则最短路树个数为 $\prod_{v \neq s} \deg_v$。

最短路计数（Luogu P1608）：DAG 路径计数。令 f_v 为 s 到 v 的最短路数量，则 $f_s = 1$，且

$$f_v = \sum_{u\,:\,\mathrm{dis}_u + w(u,v) = \mathrm{dis}_v} f_u.$$

没必要显式地把 DAG 建出来，可以在 Dijkstra / SPFA 的过程中顺便完成。

例题 9

CF 545E　Paths and Trees

给定一个无向带权图，求 s 为根的最短路树，且边权之和最小。

范围：$n, m \le 3 \cdot 10^5$

对于每个点 v，贪心选一条满足 $\mathrm{dis}_u + w(u,v) = \mathrm{dis}_v$ 且 $w(u,v)$ 最小的边即可。

差分约束

三角不等式：设 dis_u 为从给定起点 s 到 u 的最短路长度，$\mathrm{dis}_v \le \mathrm{dis}_u + w(u, v)$

差分约束：

- n 个变量：$x_1, \ldots, x_n$；
- m 个约束，每个约束形如 $x_v - x_u \le w$；
- 求一组符合条件的解。

对每个变量 x_i 建一个点 i，对约束 $x_v - x_u \le w$ 连一条 u 到 v，权值为 w 的有向边。新建一个超源 s 连接所有点，边权为 0。从 s 开始跑最短路得到 s 到所有 u 的最短路 dis_u，则 $x_u = \mathrm{dis}_u$ 为一组合法解。

差分约束

- $x_v - x_u = w$：可以分解为 $x_v - x_u \le w$ 和 $x_u - x_v \le -w$。连 $u \to v$ 权值 w，和 $v \to u$ 权值 $-w$ 两条边。
- $x_v \le w$：引入一个恒为 0 的变量 x_0，则 $x_v \le w$ 等价于 $x_v - x_0 \le w$，因此从 0 向 v 连边，边权为 w。
- $\frac{x_v}{x_u} \le w$：转化为 $\log x_v - \log x_u \le \log w$。
- 求 $x_t - x_s$ 的最大值：从 s 出发跑最短路，则 $x_t - x_s \le \mathrm{dis}_t$。
- 求 x_t 的最大值：从 0 出发跑最短路，则 $x_t \le \mathrm{dis}_t$。

例题 10

SCOI2011　糖果

有 n 个正整数变量 $x_1, \ldots, x_n$，且关于这些变量有 m 个约束。约束有 5 种形式：$x_a = x_b$，$x_a < x_b$，$x_a \leq x_b$，$x_a > x_b$，$x_a \geq x_b$，求 $\min \sum_{i=1}^{n} x_i$。

范围：$n, m \leq 10^5$

由于要求最小化 x_i，因此令 $y_i = -x_i$。引入零点 $y_0 = -1$。

- $x_i > 0$：即 $y_i - y_0 \leq 0$，连接 $0 \to i$，边权为 0。
- $x_a \leq x_b$：即 $y_b - y_a \leq 0$，连接 $a \to b$，边权为 0。
- $x_a < x_b$：即 $y_b - y_a \leq -1$，连接 $a \to b$，边权为 -1。

从 0 开始跑最短路，则 $y_i - y_0 \leq \mathrm{dis}_i$，即 $x_i \geq -\mathrm{dis}_i + 1$。

例题 10

SCOI2011　糖果

有 n 个正整数变量 $x_1, \ldots, x_n$，且关于这些变量有 m 个约束。约束有 5 种形式：$x_a = x_b$，$x_a < x_b$，$x_a \leq x_b$，$x_a > x_b$，$x_a \geq x_b$，求 $\min \sum_{i=1}^{n} x_i$。

范围：$n, m \leq 10^5$

问题转化为给一个边权为 0 或 -1 的有向图求最短路，即边权只有 01 求最长路。

如果一个强连通分量内出现 1 边则无解；否则强连通分量内全为 0 边，因此所有点距离相同。

缩点后拓扑排序在 DAG 上求最长路即可。

Thanks