最短路

李淳风

长郡中学

2024 年 9 月 27 日

单源最短路

单源最短路问题是说，给定一张有向图 $G = (V, E)$， V 是点集， E 是边集， $|V| = n, |E| = m$， 节点以 $[1, n]$ 之间的连续整数编号， (x, y, z) 描述一条从 x 出发，到达 y，长度为 z 的有向边。设 1 号点为起点，求长度为 n 的数组 $dist$，其中 $dist[i]$ 表示从起点 1 到节点 i 的最短路径的长度。

Dijkstra 算法

Dijkstra 算法的流程如下：

- 初始化 $dist[1] = 0$，其余节点的 $dist$ 值为无穷大。
- 找出一个未被标记的、$dist[x]$ 最小的节点 x，然后标记节点 x。
- 扫描节点 x 的所有出边 (x, y, z)，若 $dist[y] > dist[x] + z$，则使用 $dist[x] + z$ 更新 $dist[y]$。
- 重复上述 $2 \sim 3$ 两个步骤，知道所有节点都被标记。

Diskstra 算法基于贪心思想，它只适用于所有边的长度都是非负数的图。当边长 z 都是非负数时，全局最小值不可能再被其他节点更新，故在第 1 步中选出的节点 x 必然满足：$dist[x]$ 已经是起点到 x 的最短路径。我们不断选择全局最小值进行标记和拓展，最终可以得到起点 1 到每个节点的最短路径的长度。

该算法的时间复杂度为 $O(n^2)$，主要瓶颈在于寻找全局最小值的过程。可以用二叉堆对 $dist$ 数组进行维护，用 $O(\log m)$ 的时间获取最小值并从堆中删除，执行一条边的拓展与更新，最终可以在 $O((n+m)\log m)$ 的时间内实现 Dijkstra 算法。需要注意的是，在稠密图中 $m = O(n^2)$，此时不使用堆优化反而更优。

Bellman-Ford 算法与 SPFA 算法

给定一张有向图，若对于图中的某一条边 (x, y, z)，$dist[y] \leq dist[x] + z$ 成立，则称该边满足三角形不等式。若所有边都满足三角形不等式，则 $dist$ 数组就是所求最短路。

因此有了基于迭代思想的 Bellman-Ford 算法：

- 扫描所有边 (x, y, z)，若 $dist[y] > dis[x] + z$，则用 $dist[x] + z$ 更新 $dist[y]$。
- 重复上述步骤，直到没有更新操作发生。

Bellman-Ford 算法的时间复杂度为 $O(nm)$。

实际上，SPFA 在国际上通称为“队列优化的 Bellman-Ford”算法，仅在中国流行“SPFA”算法的称谓。SPFA 算法的流程如下：

- 建立一个队列，最初队列中只含有起点 1。
- 取出队头节点 x，扫描它的所有出边 (x, y, z)，若 $dist[y] > dist[x] + z$，则使用 $dist[x] + z$ 更新 $dist[y]$。同时，若 y 不在队列中，则把 y 入队。
- 重复上述步骤，直到队列为空。

Bellman-Ford 算法与 SPFA 算法

在任意时刻，SPFA 算法的队列都保留了待拓展的节点。每次入队相当于完成一次 $dist$ 数组的更新操作，使其满足三角形不等式。一个节点可能会入队、出队多次。最终，图中节点收敛到全部满足三角形不等式的状态。这个队列避免了 Bellman-Ford 算法中对不需要拓展的节点的冗余扫描，在随机图上运行效率为 $O(km)$ 级别，其中 k 是一个较小的常数。但在特殊构造的图上，SPFA 算法很可能退化为 $O(nm)$，因此对于 $O(nm)$ 复杂度不能通过的题目，请谨慎使用 SPFA 算法。
需要注意的是，即使图中存在长度为负数的边，Bellman-Ford 与 SPFA 算法也能够正常工作。另外，如果图中不存在长度为负数的边，那么我们可以使用优先队列来替代一般的队列，变得与堆优化 Dijkstra 算法的流程完全一致。

例题

Telephone Lines

在郊区有 N 座通信基站，P 条双向电缆，第 i 条电缆连接基站 A_i 和 B_i。特别地，1 号基站是通信公司的总站，N 号基站位于一座农场当中。现在，农场主希望对通信电路进行升级，其中升级第 i 条电缆需要划分 L_i。通信公司正在举行优惠活动。农场主可以指定一条从 1 号基站到 N 号基站的路径，并指定路径上不超过 K 条电缆，由电话公司免费提供升级服务。农场主只需要支付在该路径上剩余的电缆中，升级价格最贵的那条电缆的花费即可。求至少用多少钱能完成升级。
$0 \leq K < N \leq 1000, 1 \leq P \leq 10000$。

例题

Telephone Lines

在郊区有 N 座通信基站，P 条双向电缆，第 i 条电缆连接基站 A_i 和 B_i。特别地，1 号基站是通信公司的总站，N 号基站位于一座农场当中。现在，农场主希望对通信电路进行升级，其中升级第 i 条电缆需要划分 L_i。通信公司正在举行优惠活动。农场主可以指定一条从 1 号基站到 N 号基站的路径，并指定路径上不超过 K 条电缆，由电话公司免费提供升级服务。农场主只需要支付在该路径上剩余的电缆中，升级价格最贵的那条电缆的花费即可。求至少用多少钱能完成升级。
$0 \leq K < N \leq 1000, 1 \leq P \leq 10000$。

简单来说，本题就是在无向图上求出一条从 1 到 N 的路径，使路径上第 $K+1$ 大的边权尽量小。

Telephone Lines

我们可以仿照动态规划的思想，用 $D[x][p]$ 表示从 1 号节点到达基站 x，已经指定了 p 条电缆免费时，经过的路径上最贵的电缆的花费最小是多少。若有一条从 x 到 y 长度为 z 的边，则应该用 $\max(D[x][p], z)$ 更新 $D[y][p]$ 的最小值，用 $D[x][p]$ 更新 $D[y][p+1]$ 的最小值。前者表示不在电缆 (x, y, z) 上使用免费升级服务，后者表示使用。
虽然我们设计的状态具有“后效性”，但我们可以利用迭代思想，借助 SPFA 算法进行动态规划，直至所有状态收敛。从最短路的角度考虑，图中的节点都可以用二元组 (x, p) 表示，从 (x, p) 到 (y, p) 有长度为 z 的边，从 (x, p) 到 $(y, p+1)$ 有长度为 0 的边，$D[x][p]$ 表示从起点 $(1, 0)$ 到节点 (x, p)，路径上最长的边最短是多少。这是 $N * K$ 个点，$P * K$ 条边的广义最短路问题，被称为分层图最短路径。我们可以枚举层数 p，每次在一层图中跑最短路，然后再进入下一层图。使用堆优化的 Dijkstra 的话，复杂度为 $O(KP \log P)$，不太推荐大家使用。

Telephone Lines

注意到本题的答案显然具有单调性，因为支付的钱更多时，合法的升级方案一定包含了花费更少的升级方案。所以，我们可以二分答案，把问题转化为：是否存在一种合法的升级方法，使花费不超过 mid。转化后的判定问题非常容易，只需把价格大于 mid 的电缆看作长度为 1 的边，把升级价格不超过 mid 的电缆看作长度为 0 的边，然后判断从 1 到 N 的最短路是否不超过 K 即可。对于这种边权只有 0 和 1 的最短路问题，我们可以使用双端队列 BFS 来解决。
时间复杂度为 $O((N+P)\log MAX_L)$。

例题

最优贸易

C 国有 n 个大城市和 m 条道路，每条道路连接这 n 个城市中的某两个城市。任意两个城市之间最多只有一条道路直接相连。这 m 条道路中有一部分为单向通行的道路，一部分为双向通行的道路，双向通行的道路在统计条数时也计为 1 条。
C 国幅员辽阔，各地的资源分布情况各不相同，这就导致了同一种商品在不同城市的价格不一定相同。但是，同一种商品在同一个城市的买入价和卖出价始终是相同的。
商人阿龙来到 C 国旅游。当他得知同一种商品在不同城市的价格可能会不同这一信息之后，便决定在旅游的同时，利用商品在不同城市中的差价赚回一点旅费。设 C 国 n 个城市的标号为 $1 \sim n$，阿龙决定从 1 号城市出发，并最终在 n 号城市结束自己的旅行。在旅游的过程中，任何城市可以重复经过多次，但不要求经过所有 n 个城市。阿龙通过这样的贸易方式赚取旅费：他会选择一个经过的城市买入他最喜欢的商品 水晶球，并在之后经过的另一个城市卖出这个水晶球，用赚取的差价当做旅费。由于阿龙主要是来 C 国旅游，他决定这个贸易只进行最多一次，当然，在赚不到差价的情况下他就无需进行贸易。
现在给出 n 个城市的水晶球价格，m 条道路的信息（每条道路所连接的两个城市的编号以及该条道路的通行情况）。请你告诉阿龙，他最多能赚取多少旅费。
$n \leq 100000, m \leq 500000$。

最优贸易

本题是让我们在一张节点带有权值的图上找出一条从 1 到 n 的路径，使路径上能选出两个点 p, q（先经过 p 后经过 q），并且 "节点 p 的权值减去节点 q 的权值" 最大。

首先，图中双向同行的道路可以看作两条方向相反的单向通行道路，因此我们只需把这张图当作有向图考虑即可。除此之外，我们再建立一张反图（在原图基础上把所有边取反后）。

我们先以 1 为起点跑一遍最短路，求出一个数组 D，其中 $D[x]$ 表示从节点 1 到节点 x 的所有路径中，能经过的权值最小的节点的权值。D 数组的计算过程与最短路的计算过程类似，只需把 "用 $D[x] + w(x, y)$ 更新 $D[y]$" 改为 "用 $\min(D[x], price[y])$ 更新 $D[y]$" 即可。注意上面的更新不满足 Dijkstra 算法的贪心性质。因此在这里我们可以把所有节点按照权值从小到大排序，初始把权值最小的点加入队列，每次从队首取出点开始 BFS，把能到达的节点的 D 数组更新并加入队列。若队列为空，则接着按照权值从小到大枚举节点，如果还没有加入过队列则加入队列并继续 BFS。

再以 n 为起点，在反图上求出一个数组 F，其中 $F[x]$ 表示在原图上从节点 x 到节点 n（在反图上从 n 到 x）的所有路径中，能够经过的节点中最大权值。最后枚举每个节点 x，用 $F[x] - D[x]$ 更新答案即可。

例题

道路与航线

农夫 John 正在一个新的销售区域对他的牛奶销售方案进行调查。他想把牛奶送到 T 个城镇（$1 \leq T \leq 25000$），编号为 1 到 T 。这些城镇之间通过 R 条道路 ($1 \leq R \leq 50000$，编号为 1 到 R) 和 P 条航线 ($1 \leq P \leq 50000$，编号为 1 到 P) 连接。每条道路 i 或者航线 i 连接城镇 A_i 与 B_i，花费为 C_i。
对于道路 $0 \leq C_i \leq 10000$; 然而航线的花费很神奇，花费 C_i 可能是负数 ($-10000 \leq C_i \leq 10000$)。道路是双向的，可以从 A_i 到 B_i，也可以从 B_i 到 A_i ，花费都是 C_i 。然而航线与之不同，只可以从 A_i 到 B_i 。事实上，由于最近恐怖主义太嚣张，为了社会和谐，出台了一些政策保证：如果有一条航线可以从 A_i 到 B_i ，那么保证不可能通过一些道路和航线从 B_i 回到 A_i 。由于农夫 John 的牛奶公认十分给力，他需要运送牛奶到每一个城镇。他想找到从发送中心城镇 S 把奶牛送到每个城镇的最便宜的方案，或者知道这是不可能的。

道路与航线

本题是一道明显的单源最短路问题，但图中带有负权边，不能使用 Dijkstra 算法。至于 SPFA 算法，在最坏情况下的复杂度无法接受时，我们一般不做考虑。注意到题目中有一个特殊性质——双向边都是非负的，只有单向边可能是负的，并且单向边不构成环。我们可以尝试利用这个性质来解答。

如果只把双向边添加到图中，那么会形成若干个连通块。若把每个连通块看作一个点，再把单向边添加到图中，会得到一张有向无环图。而在有向无环图上，无论边权正负，都可以按照拓扑序进行扫描，在线性时间内求出单源最短路。这启发我们用拓扑排序的框架处理整个图，但在双向边构成的每个连通块内部使用堆优化的 Dijkstra 算法快速计算该块内的最短路信息。

整个算法的复杂度为 $O(T + P + R\log T)$。

任意两点间最短路径

为了求出图中任意两点间的最短路径，当然可以把每个点作为起点，求解 N 次单源最短路问题。不过，在任意两点间最短路问题中，图一般比较稠密，使用 Floyd 算法可以在 $O(N^3)$ 的时间内完成求解，并且程序实现非常简单。

设 $D[k][i][j]$ 表示只经过 $1 \sim k$ 这 k 个节点，从 i 到 j 的最短路长度。该问题可以划分为两个子问题，即从 i 先到 k 再到 j，或者不经过 k:

$$D[k][i][j] = \min(D[k-1][i][j], D[k-1][i][k] + D[i-1][k][j])$$

可以看到，Floyd 算法的本质是动态规划，k 是阶段，所以必须先枚举 k。i, j 是附加状态，所以放在内层循环。

与背包问题类似，k 这一维可以被省略。为了方便代码实现，我们还可以在一开始就用 D 保存邻接矩阵，然后枚举 k 进行动态规划:

$$D[i][j] = \min(D[i][j], D[i][k] + D[k][j])$$

最终 $D[i][j]$ 就保存了 i 到 j 的最短路长度。

传递闭包

在交际网络中，给出若干个元素和若干对二元关系，并且关系具有传递性。“通过传递性推导出尽量多的元素之间的关系”的问题被称为传递闭包。
建立邻接矩阵 D，其中 $D[i][j] = 1$ 表示 i 与 j 有关系，$D[i][j] = 0$ 表示 i 与 j 没有关系。特别地，$D[i][i]$ 始终为 1。使用 Floyd 算法可以解决闭包问题。由于是 bool 类型的数组，还可以使用 bitset 加速。

例题

Sorting It All Out

给定 n 个变量，m 个不等式。不等式之间具有传递性，即若 $A > B$ 且 $B > C$，则有 $A > C$。现在请你判断这 m 个不等式是否有矛盾。若无矛盾，则判断这 m 个不等式是否能确定每一对变量之间的关系。若能，则求出 t 的最小值，满足仅用前 t 个不等式就能确定每一对变量之间的大小关系。$2 \leq n \leq 26$。

例题

Sorting It All Out

给定 n 个变量，m 个不等式。不等式之间具有传递性，即若 $A > B$ 且 $B > C$，则有 $A > C$。现在请你判断这 m 个不等式是否有矛盾。若无矛盾，则判断这 m 个不等式是否能确定每一对变量之间的关系。若能，则求出 t 的最小值，满足仅用前 t 个不等式就能确定每一对变量之间的大小关系。$2 \leq n \leq 26$。

这是一道"有向"的传递闭包的问题。对于每个形如 $i < j$ 的不等式，令 $D[i][j] = 1$。对于 $i > j$ 的不等式，则看作 $j < i$ 进行处理。对于其它情况，均有 $D[i][j] = 0$。
使用 Floyd 算法对 D 进行传递闭包。传递闭包完成后，若存在变量 i, j 使得 $D[i][j], D[j][i]$ 均为 1，则说明存在矛盾。若存在 i, j 使得 $D[i][j], D[j][i]$ 均为 0，则说明不能确定 i 和 j 之间的大小关系。如果对于每一对变量 i, j，$D[i][j]$ 与 $D[j][i]$ 有且仅有一个为 1，则说明每队变量之间的大小关系都已经被确定。
因此，我们二分或枚举 t，每次通过 Floyd 算法进行判断即可。

例题

Sightseeing Trip

给定一张无向图，求图中一个至少包含 3 个点的环，环上的节点不重复，并且环上的边的长度之和最小。该问题称为无向图的最小环问题。在本题中，你需要输出最小环的方案，若最小环不唯一，输出任意一个即可。图的节点数不超过 100。

例题

Sightseeing Trip

给定一张无向图，求图中一个至少包含 3 个点的环，环上的节点不重复，并且环上的边的长度之和最小。该问题称为无向图的最小环问题。在本题中，你需要输出最小环的方案，若最小环不唯一，输出任意一个即可。图的节点数不超过 100。

考虑 Floyd 算法的过程。当外层循环枚举到 $k = k'$ 时，$D[i][j]$ 保存着 "经过编号不超过 $k-1$ 的节点" 从 i 到 j 的最短路长度。
于是我们考虑编号不超过 k' 的、经过节点 k' 的环，其长度可以在更新 D 之前用 $D[i][j] + a[j][k'] + a[k'][i]$ 来描述。这个式子中的 i, j 就是环上与 k' 相邻的两个节点。枚举 k 并统计上式的最小值即可得到全局的最小值。

例题

Sightseeing Trip

给定一张无向图，求图中一个至少包含 3 个点的环，环上的节点不重复，并且环上的边的长度之和最小。该问题称为无向图的最小环问题。在本题中，你需要输出最小环的方案，若最小环不唯一，输出任意一个即可。图的节点数不超过 100。

考虑 Floyd 算法的过程。当外层循环枚举到 $k = k'$ 时，$D[i][j]$ 保存着 “经过编号不超过 $k-1$ 的节点” 从 i 到 j 的最短路长度。
于是我们考虑编号不超过 k' 的、经过节点 k' 的环，其长度可以在更新 D 之前用 $D[i][j] + a[j][k'] + a[k'][i]$ 来描述。这个式子中的 i, j 就是环上与 k' 相邻的两个节点。枚举 k 并统计上式的最小值即可得到全局的最小值。
对于有向图的最小环问题，可以枚举起点 $s = 1 \sim n$，执行堆优化的 Dijkstra 算法求解单源最短路径。s 一定是第一个被从堆中取出的节点，我们扫描 s 的所有出边，拓展、更新完成后令 $d[s] = +\infty$，然后继续求解。当 s 第二次被从堆中取出时，$d[s]$ 就是经过 s 的最小环长度。

例题

Cow Relays

给定一张由 $T(2 \le T \le 100)$ 条边构成的无向图，点的编号为 $1 \sim 1000$ 之间的整数。求从起点 S 到终点 E 恰好经过 $N(2 \le N \le 10^6)$ 条边（可以重复经过）的最短路。

例题

Cow Relays

给定一张由 $T(2 \le T \le 100)$ 条边构成的无向图，点的编号为 $1 \sim 1000$ 之间的整数。求从起点 S 到终点 E 恰好经过 $N(2 \le N \le 10^6)$ 条边（可以重复经过）的最短路。

首先，由于点数比边数要多，我们可以先对点的编号进行离散化，找出 $P(P \le 2T)$ 个有用的点。之后，我们设该图的邻接矩阵为 A，考虑以下方程：

$$\forall i.j \in [1, P], \quad B[i][j] = \min_{1 \le k \le P} \{A[i][k] + A[k][j]\}$$

我们可以认为 $A[i][j]$ 表示从 i 到 j 经过一条边的最短路。在上面的式子中，我们对于每队二元组 (i, j) 枚举了中转点 k，从 i 先到 k，再从 k 到 j。因此，$B[i][j]$ 表示从 i 到 j 经过两条边的最短路。

Cow Relays

一般地，若矩阵 A^m 保存任意两点之间恰好经过 m 条边的最短路，则：

$$\forall i,j \in [1,P] \quad (A^{r+m})[i][j] = \min_{1\le k\le P}(A^r)[i][k] + (A^m)[k][j]$$

回顾之前学过的矩阵乘法相关内容，我们发现上式其实等价于一个关于 $\min$ 与加法运算的广义矩阵乘法。而这个广义的矩阵乘法也满足结合律。因此，我们可以用快速幂计算出 A^N，具体来说就是把原本矩阵乘法时的乘法变为加法，原本的加法用 $\min$ 运算代替。
最后，$(A^N)[S,E]$ 即为本题的答案。时间复杂度为 $O(T^3 \log N)$。