形形色色的最短路

Claris

Hangzhou Dianzi University

2017 年 8 月 5 日

Overview

- 最短路问题是经典的图论问题，一般求解最短路有 BFS 算法、Floyd 算法、Dijkstra 算法、Bellman-Ford 算法等。

Overview

- 最短路问题是经典的图论问题，一般求解最短路有 BFS 算法、Floyd 算法、Dijkstra 算法、Bellman-Ford 算法等。
- 有些问题中图过于庞大，不能直接建出，有些问题则是隐式的最短路。

Overview

- 最短路问题是经典的图论问题，一般求解最短路有 BFS 算法、Floyd 算法、Dijkstra 算法、Bellman-Ford 算法等。
- 有些问题中图过于庞大，不能直接建出，有些问题则是隐式的最短路。
- 解决这些问题的第一个方法是优化建图。

冰原探险

平面上有 n 个平行于坐标轴的矩形冰山，冰山之间不会相互接触。

冰原探险

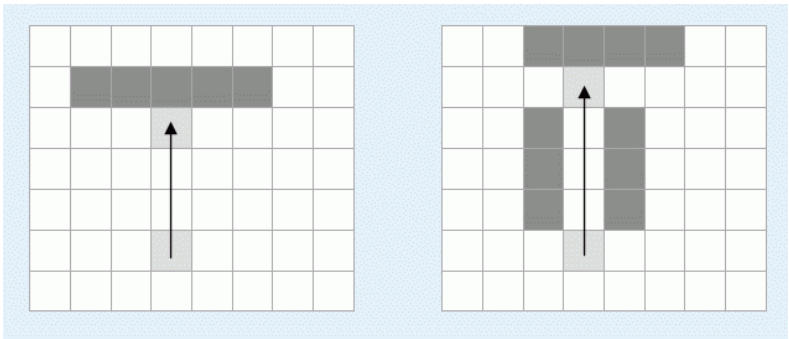
平面上有 n 个平行于坐标轴的矩形冰山，冰山之间不会相互接触。

你需要控制冰块往上下左右滑动，直到撞到某个冰山或到达目的地。求最小滑动次数。

冰原探险

平面上有 n 个平行于坐标轴的矩形冰山，冰山之间不会相互接触。

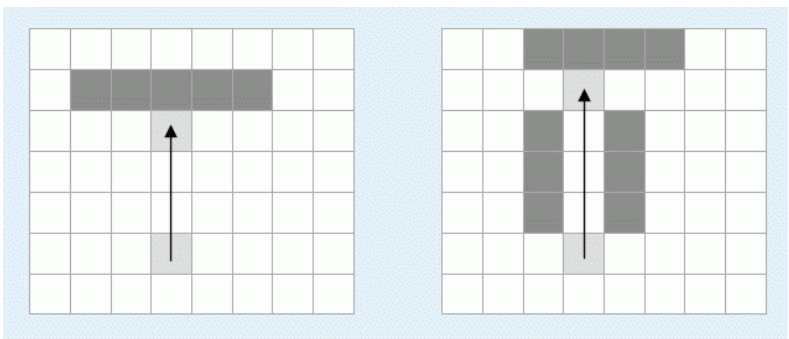
你需要控制冰块往上下左右滑动，直到撞到某个冰山或到达目的地。求最小滑动次数。



冰原探险

平面上有 n 个平行于坐标轴的矩形冰山，冰山之间不会相互接触。

你需要控制冰块往上下左右滑动，直到撞到某个冰山或到达目的地。求最小滑动次数。

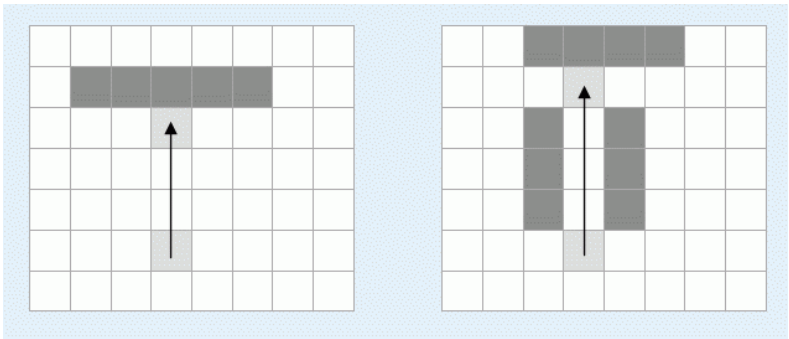


- $1 \leq n \leq 100000, 1 \leq x_i, y_i \leq 10^9$。

冰原探险

平面上有 n 个平行于坐标轴的矩形冰山，冰山之间不会相互接触。

你需要控制冰块往上下左右滑动，直到撞到某个冰山或到达目的地。求最小滑动次数。



- $1 \leq n \leq 100000, 1 \leq x_i, y_i \leq 10^9$。
- Source： CTSC 2000

Solution

- 有效状态只有某个矩形的某条边上以及起点和终点，共 $4n+2$ 个。

Solution

- 有效状态只有某个矩形的某条边上以及起点和终点，共 $4n+2$ 个。
- 从四个方向分别扫描线，用 set 维护扫描线之前最近的矩形，以此求出转移图。

Solution

- 有效状态只有某个矩形的某条边上以及起点和终点，共 $4n+2$ 个。
- 从四个方向分别扫描线，用 set 维护扫描线之前最近的矩形，以此求出转移图。
- 建好图之后就可以 BFS 求出答案。

Solution

- 有效状态只有某个矩形的某条边上以及起点和终点，共 $4n+2$ 个。
- 从四个方向分别扫描线，用 set 维护扫描线之前最近的矩形，以此求出转移图。
- 建好图之后就可以 BFS 求出答案。
- 时间复杂度 $O(n \log n)$。

Walk

给定一张 n 个点的有向图，点 i 的权值为 val_i，若 $val_i \text{ and } val_j = val_j$，则 i 向 j 连单向边。

Walk

给定一张 n 个点的有向图，点 i 的权值为 val_i，若 $val_i \ and \ val_j = val_j$，则 i 向 j 连单向边。

另外再给定 m 条有向边，求 1 到每个点最少经过的边数。

Walk

给定一张 n 个点的有向图，点 i 的权值为 val_i，若 $val_i \ and \ val_j = val_j$，则 i 向 j 连单向边。

另外再给定 m 条有向边，求 1 到每个点最少经过的边数。

- $2 \leq n \leq 200000$。

Walk

给定一张 n 个点的有向图，点 i 的权值为 val_i，若 $val_i \text{ and } val_j = val_j$，则 i 向 j 连单向边。

另外再给定 m 条有向边，求 1 到每个点最少经过的边数。

- $2 \leq n \leq 200000$。
- $0 \leq m \leq 300000$。

Walk

给定一张 n 个点的有向图，点 i 的权值为 val_i，若 $val_i \text{ and } val_j = val_j$，则 i 向 j 连单向边。

另外再给定 m 条有向边，求 1 到每个点最少经过的边数。

- $2 \leq n \leq 200000$。
- $0 \leq m \leq 300000$。
- $0 \leq val_i < 2^{20}$。

Walk

给定一张 n 个点的有向图，点 i 的权值为 val_i，若 $val_i \ and \ val_j = val_j$，则 i 向 j 连单向边。

另外再给定 m 条有向边，求 1 到每个点最少经过的边数。

- $2 \leq n \leq 200000$。
- $0 \leq m \leq 300000$。
- $0 \leq val_i < 2^{20}$。
- Source：BZOJ 2016 年 NOIP 十连测第一场

Solution

- 考虑新增 2^{20} 个点，这些点中 i 向去掉某一位的 1 的点连一条权值为 0 的有向边。

Solution

- 考虑新增 2^{20} 个点，这些点中 i 向去掉某一位的 1 的点连一条权值为 0 的有向边。
- 对于原来的 n 个点，先把 m 条边连好，然后对于 i 号点，由它向新增的第 val_i 个点连一条权值为 1 的有向边，再由新增的第 val_i 个点向它连一条权值为 0 的有向边。

Solution

- 考虑新增 2^{20} 个点，这些点中 i 向去掉某一位的 1 的点连一条权值为 0 的有向边。
- 对于原来的 n 个点，先把 m 条边连好，然后对于 i 号点，由它向新增的第 val_i 个点连一条权值为 1 的有向边，再由新增的第 val_i 个点向它连一条权值为 0 的有向边。
- BFS 的时候，每次要把用 0 权值的边连接的所有点都加入队尾，以保证距离不降。

Solution

- 考虑新增 2^{20} 个点，这些点中 i 向去掉某一位的 1 的点连一条权值为 0 的有向边。
- 对于原来的 n 个点，先把 m 条边连好，然后对于 i 号点，由它向新增的第 val_i 个点连一条权值为 1 的有向边，再由新增的第 val_i 个点向它连一条权值为 0 的有向边。
- BFS 的时候，每次要把用 0 权值的边连接的所有点都加入队尾，以保证距离不降。
- 时间复杂度 $O(20 \cdot 2^{20} + n + m)$。

Bus Trip

有 n 座城镇，m 条巴士单行线。你在 0 时刻位于 1 号城镇，想要通过巴士在 T 时刻到达 n 号城镇。

Bus Trip

有 n 座城镇，m 条巴士单行线。你在 0 时刻位于 1 号城镇，想要通过巴士在 T 时刻到达 n 号城镇。

对于第 i 条巴士路线，它会在时间 $[a_i, b_i]$ 内离开城镇 s_i，在时间 $[c_i, d_i]$ 到达城镇 t_i。

Bus Trip

有 n 座城镇，m 条巴士单行线。你在 0 时刻位于 1 号城镇，想要通过巴士在 T 时刻到达 n 号城镇。

对于第 i 条巴士路线，它会在时间 $[a_i, b_i]$ 内离开城镇 s_i，在时间 $[c_i, d_i]$ 到达城镇 t_i。

你每次换乘巴士时，都要保证当前巴士的最晚到达时间不迟于下一辆巴士的最早出发时间。

Bus Trip

有 n 座城镇，m 条巴士单行线。你在 0 时刻位于 1 号城镇，想要通过巴士在 T 时刻到达 n 号城镇。

对于第 i 条巴士路线，它会在时间 $[a_i, b_i]$ 内离开城镇 s_i，在时间 $[c_i, d_i]$ 到达城镇 t_i。

你每次换乘巴士时，都要保证当前巴士的最晚到达时间不迟于下一辆巴士的最早出发时间。

请找一条合法的路线，使得最坏情况下总等待时间最小。

Bus Trip

有 n 座城镇，m 条巴士单行线。你在 0 时刻位于 1 号城镇，想要通过巴士在 T 时刻到达 n 号城镇。

对于第 i 条巴士路线，它会在时间 $[a_i, b_i]$ 内离开城镇 s_i，在时间 $[c_i, d_i]$ 到达城镇 t_i。

你每次换乘巴士时，都要保证当前巴士的最晚到达时间不迟于下一辆巴士的最早出发时间。

请找一条合法的路线，使得最坏情况下总等待时间最小。

- $1 \leq n \leq 50000, 1 \leq m \leq 100000$。

Bus Trip

有 n 座城镇，m 条巴士单行线。你在 0 时刻位于 1 号城镇，想要通过巴士在 T 时刻到达 n 号城镇。

对于第 i 条巴士路线，它会在时间 $[a_i, b_i]$ 内离开城镇 s_i，在时间 $[c_i, d_i]$ 到达城镇 t_i。

你每次换乘巴士时，都要保证当前巴士的最晚到达时间不迟于下一辆巴士的最早出发时间。

请找一条合法的路线，使得最坏情况下总等待时间最小。

- $1 \leq n \leq 50000, 1 \leq m \leq 100000$。
- $0 \leq a_i \leq b_i < c_i \leq d_i \leq 10^9, 0 \leq T \leq 10^9$。

Bus Trip

有 n 座城镇，m 条巴士单行线。你在 0 时刻位于 1 号城镇，想要通过巴士在 T 时刻到达 n 号城镇。

对于第 i 条巴士路线，它会在时间 $[a_i, b_i]$ 内离开城镇 s_i，在时间 $[c_i, d_i]$ 到达城镇 t_i。

你每次换乘巴士时，都要保证当前巴士的最晚到达时间不迟于下一辆巴士的最早出发时间。

请找一条合法的路线，使得最坏情况下总等待时间最小。

- $1 \leq n \leq 50000, 1 \leq m \leq 100000$。
- $0 \leq a_i \leq b_i < c_i \leq d_i \leq 10^9, 0 \leq T \leq 10^9$。
- Source : Baltic Olympiad In Informatics 2005

Solution

- 建立新图，原图中每条边在新图中是点，新图中每个点的点权为 $-c_i + b_i$，边权均为 0。

Solution

- 建立新图，原图中每条边在新图中是点，新图中每个点的点权为 $-c_i + b_i$，边权均为 0。
- 若 $d_i \leq a_j$，则连一条 i 到 j 的单向边。

Solution

- 建立新图，原图中每条边在新图中是点，新图中每个点的点权为 $-c_i + b_i$，边权均为 0。
- 若 $d_i \leq a_j$，则连一条 i 到 j 的单向边。
- 在新图中求出最短路，最后将答案加上 T 即可。

Solution

- 建立新图，原图中每条边在新图中是点，新图中每个点的点权为 $-c_i + b_i$，边权均为 0。
- 若 $d_i \leq a_j$，则连一条 i 到 j 的单向边。
- 在新图中求出最短路，最后将答案加上 T 即可。
- 注意到新图是个 DAG，因此可以记搜求解。

Solution

- 建立新图，原图中每条边在新图中是点，新图中每个点的点权为 $-c_i + b_i$，边权均为 0。
- 若 $d_i \leq a_j$，则连一条 i 到 j 的单向边。
- 在新图中求出最短路，最后将答案加上 T 即可。
- 注意到新图是个 DAG，因此可以记搜求解。
- 边数 $O(m^2)$，不能接受。

Solution

- 对于原图中每个点，将所有入边和出边分别排序，建立一排虚点表示后缀。

Solution

- 对于原图中每个点，将所有入边和出边分别排序，建立一排虚点表示后缀。
- 那么每条边会向虚点中某个点连边，表示向后缀连边。

Solution

- 对于原图中每个点，将所有入边和出边分别排序，建立一排虚点表示后缀。
- 那么每条边会向虚点中某个点连边，表示向后缀连边。
- 通过双指针将边数优化至 $O(m)$。

Solution

- 对于原图中每个点，将所有入边和出边分别排序，建立一排虚点表示后缀。
- 那么每条边会向虚点中某个点连边，表示向后缀连边。
- 通过双指针将边数优化至 $O(m)$。
- 时间复杂度 $O(m \log m)$。

天才黑客

有一棵 k 个点的字典树和一个 n 个点，m 条边的有向图 G。

天才黑客

有一棵 k 个点的字典树和一个 n 个点，m 条边的有向图 G。

每条边有一个权值 w_i，同时上面还有个字符串 str_i，用字典树上根到某个点的路径表示。

天才黑客

有一棵 k 个点的字典树和一个 n 个点，m 条边的有向图 G。

每条边有一个权值 w_i，同时上面还有个字符串 str_i，用字典树上根到某个点的路径表示。

定义一条路径的代价为经过的所有边的权值之和，加上相邻两条边上字符串的最长公共前缀的长度。

天才黑客

有一棵 k 个点的字典树和一个 n 个点，m 条边的有向图 G。

每条边有一个权值 w_i，同时上面还有个字符串 str_i，用字典树上根到某个点的路径表示。

定义一条路径的代价为经过的所有边的权值之和，加上相邻两条边上字符串的最长公共前缀的长度。

求 1 号点到每个点的代价最小的路径长度。

天才黑客

有一棵 k 个点的字典树和一个 n 个点，m 条边的有向图 G。

每条边有一个权值 w_i，同时上面还有个字符串 str_i，用字典树上根到某个点的路径表示。

定义一条路径的代价为经过的所有边的权值之和，加上相邻两条边上字符串的最长公共前缀的长度。

求 1 号点到每个点的代价最小的路径长度。

- $1 \leq n, m \leq 50000$。

天才黑客

有一棵 k 个点的字典树和一个 n 个点，m 条边的有向图 G。

每条边有一个权值 w_i，同时上面还有个字符串 str_i，用字典树上根到某个点的路径表示。

定义一条路径的代价为经过的所有边的权值之和，加上相邻两条边上字符串的最长公共前缀的长度。

求 1 号点到每个点的代价最小的路径长度。

- $1 \leq n, m \leq 50000$。
- $1 \leq k \leq 20000$。

天才黑客

有一棵 k 个点的字典树和一个 n 个点，m 条边的有向图 G。

每条边有一个权值 w_i，同时上面还有个字符串 str_i，用字典树上根到某个点的路径表示。

定义一条路径的代价为经过的所有边的权值之和，加上相邻两条边上字符串的最长公共前缀的长度。

求 1 号点到每个点的代价最小的路径长度。

- $1 \leq n, m \leq 50000$。
- $1 \leq k \leq 20000$。
- $0 \leq w_i \leq 20000$。

天才黑客

有一棵 k 个点的字典树和一个 n 个点，m 条边的有向图 G。

每条边有一个权值 w_i，同时上面还有个字符串 str_i，用字典树上根到某个点的路径表示。

定义一条路径的代价为经过的所有边的权值之和，加上相邻两条边上字符串的最长公共前缀的长度。

求 1 号点到每个点的代价最小的路径长度。

- $1 \leq n, m \leq 50000$。
- $1 \leq k \leq 20000$。
- $0 \leq w_i \leq 20000$。
- Source： SDOI 2017

Solution

- 同样地，建立新图，原图中每条边在新图中是点，点权为 w_i，边权为两个字符串的 LCP。

Solution

- 同样地，建立新图，原图中每条边在新图中是点，点权为 w_i，边权为两个字符串的 LCP。
- 对字典树进行 DFS，将每个点周围一圈边对应的字符串按 DFS 序从小到大排序。

Solution

- 同样地，建立新图，原图中每条边在新图中是点，点权为 w_i，边权为两个字符串的 LCP。
- 对字典树进行 DFS，将每个点周围一圈边对应的字符串按 DFS 序从小到大排序。
- 令 $h_i = LCP(str_i, str_{i+1})$，则 $LCP(str_l, str_r) = \min(h_{l..r-1})$。

Solution

- 同样地，建立新图，原图中每条边在新图中是点，点权为 w_i，边权为两个字符串的 LCP。
- 对字典树进行 DFS，将每个点周围一圈边对应的字符串按 DFS 序从小到大排序。
- 令 $h_i = LCP(str_i, str_{i+1})$，则 $LCP(str_l, str_r) = \min(h_{l..r-1})$。
- 枚举每个 h_i 作为分界线，那么新图中两侧的点均可以通过不超过 h_i 的代价互相访问。

Solution

- 同样地，建立新图，原图中每条边在新图中是点，点权为 w_i，边权为两个字符串的 LCP。
- 对字典树进行 DFS，将每个点周围一圈边对应的字符串按 DFS 序从小到大排序。
- 令 $h_i = LCP(str_i, str_{i+1})$，则 $LCP(str_l, str_r) = \min(h_{l..r-1})$。
- 枚举每个 h_i 作为分界线，那么新图中两侧的点均可以通过不超过 h_i 的代价互相访问。
- 建立一排前缀虚点和后缀虚点然后对应前后缀之间连边即可。

Solution

- 同样地，建立新图，原图中每条边在新图中是点，点权为 w_i，边权为两个字符串的 LCP。
- 对字典树进行 DFS，将每个点周围一圈边对应的字符串按 DFS 序从小到大排序。
- 令 $h_i = LCP(str_i, str_{i+1})$，则 $LCP(str_l, str_r) = \min(h_{l..r-1})$。
- 枚举每个 h_i 作为分界线，那么新图中两侧的点均可以通过不超过 h_i 的代价互相访问。
- 建立一排前缀虚点和后缀虚点然后对应前后缀之间连边即可。
- 时间复杂度 $O(m \log m)$。

Summary

- 以上问题均是通过利用题目性质来优化建图。

Summary

- 以上问题均是通过利用题目性质来优化建图。
- 其它一些问题中，可以通过往图中嵌入数据结构来实现形如“向某个区间连边”的功能。

Summary

- 以上问题均是通过利用题目性质来优化建图。
- 其它一些问题中，可以通过往图中嵌入数据结构来实现形如“向某个区间连边”的功能。
- 这种方法往往非常暴力，会导致复杂度多一个 log，有些时候甚至无法按照题目要求顺利建图。

Dijkstra 算法的本质

- Dijkstra 算法是针对非负权图的一种单源最短路算法。

Dijkstra 算法的本质

- Dijkstra 算法是针对非负权图的一种单源最短路算法。
- 它的原理是从源点开始，每次寻找最近的未访问过的点，标记其最短路，然后松弛与其相邻的点。

Dijkstra 算法的本质

- Dijkstra 算法是针对非负权图的一种单源最短路算法。
- 它的原理是从源点开始，每次寻找最近的未访问过的点，标记其最短路，然后松弛与其相邻的点。
- 每个点一旦被标记过，就没有再次被标记的需要，因为距离单调不降。

Dijkstra 算法的本质

- Dijkstra 算法是针对非负权图的一种单源最短路算法。
- 它的原理是从源点开始，每次寻找最近的未访问过的点，标记其最短路，然后松弛与其相邻的点。
- 每个点一旦被标记过，就没有再次被标记的需要，因为距离单调不降。
- 利用每个点只会访问一次这一特性，我们可以设计算法来忽略那些已经访问过的点。

补图最短路

给定一张 n 个点的完全无向图，上面删掉了 m 条边。

补图最短路

给定一张 n 个点的完全无向图，上面删掉了 m 条边。

求 1 号点到每个点最少经过的边数。

补图最短路

给定一张 n 个点的完全无向图，上面删掉了 m 条边。求 1 号点到每个点最少经过的边数。

- $2 \leq n \leq 1000000$。

补图最短路

给定一张 n 个点的完全无向图，上面删掉了 m 条边。求 1 号点到每个点最少经过的边数。

- $2 \leq n \leq 1000000$。
- $0 \leq m \leq 3000000$。

Solution

- 因为没有边权，所以直接 BFS 即可。

Solution

- 因为没有边权，所以直接 BFS 即可。
- 对于当前点 x，我们需要访问所有与 x 有边相连的点，但要忽略已经经过的点。

Solution

- 因为没有边权，所以直接 BFS 即可。
- 对于当前点 x，我们需要访问所有与 x 有边相连的点，但要忽略已经经过的点。
- 用链表维护所有未访问的点，遍历整个链表，若与 x 有边相连则将其删除。

Solution

- 因为没有边权，所以直接 BFS 即可。
- 对于当前点 x，我们需要访问所有与 x 有边相连的点，但要忽略已经经过的点。
- 用链表维护所有未访问的点，遍历整个链表，若与 x 有边相连则将其删除。
- 每个点只会被删除一次，或者在 m 条边中被跳过。

Solution

- 因为没有边权，所以直接 BFS 即可。
- 对于当前点 x，我们需要访问所有与 x 有边相连的点，但要忽略已经经过的点。
- 用链表维护所有未访问的点，遍历整个链表，若与 x 有边相连则将其删除。
- 每个点只会被删除一次，或者在 m 条边中被跳过。
- 时间复杂度 $O(n+m)$。

Subway

在地铁系统中有 S 个站点，n 条双向直线线路。

Subway

在地铁系统中有 S 个站点，n 条双向直线线路。

每换乘一次就要支付 1 元钱 (包括换方向)，相邻两站之间的时间为 1 分钟。

Subway

在地铁系统中有 S 个站点，n 条双向直线线路。

每换乘一次就要支付 1 元钱 (包括换方向)，相邻两站之间的时间为 1 分钟。

求 1 号站点到 S 号站点在费用最小的情况下最多可以乘坐多少时间地铁。

Subway

在地铁系统中有 S 个站点，n 条双向直线线路。

每换乘一次就要支付 1 元钱 (包括换方向)，相邻两站之间的时间为 1 分钟。

求 1 号站点到 S 号站点在费用最小的情况下最多可以乘坐多少时间地铁。

- $2 \leq S \leq 300000$。

Subway

在地铁系统中有 S 个站点，n 条双向直线线路。

每换乘一次就要支付 1 元钱 (包括换方向)，相邻两站之间的时间为 1 分钟。

求 1 号站点到 S 号站点在费用最小的情况下最多可以乘坐多少时间地铁。

- $2 \leq S \leq 300000$。
- $1 \leq n \leq 100000$。

Subway

在地铁系统中有 S 个站点，n 条双向直线线路。

每换乘一次就要支付 1 元钱 (包括换方向)，相邻两站之间的时间为 1 分钟。

求 1 号站点到 S 号站点在费用最小的情况下最多可以乘坐多少时间地铁。

- $2 \leq S \leq 300000$。
- $1 \leq n \leq 100000$。
- $\sum length(i) \leq 10^6$。

Subway

在地铁系统中有 S 个站点，n 条双向直线线路。

每换乘一次就要支付 1 元钱 (包括换方向)，相邻两站之间的时间为 1 分钟。

求 1 号站点到 S 号站点在费用最小的情况下最多可以乘坐多少时间地铁。

- $2 \leq S \leq 300000$。
- $1 \leq n \leq 100000$。
- $\sum length(i) \leq 10^6$。
- Source： CERC 2013

Solution

- 通过 BFS 可以求出到每个站点的最小花费。

Solution

- 通过 BFS 可以求出到每个站点的最小花费。
- 每次从队首取出一个站点，枚举所有它能花费 1 块钱就到达的线路，通过两遍递推求出对应线路内部的最大时间。

Solution

- 通过 BFS 可以求出到每个站点的最小花费。
- 每次从队首取出一个站点，枚举所有它能花费 1 块钱就到达的线路，通过两遍递推求出对应线路内部的最大时间。
- 注意到每个点和每条线路只有第一次使用时有用，故一旦使用过就不再重复计算。

Solution

- 通过 BFS 可以求出到每个站点的最小花费。
- 每次从队首取出一个站点，枚举所有它能花费 1 块钱就到达的线路，通过两遍递推求出对应线路内部的最大时间。
- 注意到每个点和每条线路只有第一次使用时有用，故一旦使用过就不再重复计算。
- 时间复杂度 $O(S + n + \sum length(i))$。

国王的命令

给定一棵 n 个点的无根树，树边的长度都是 1。

国王的命令

给定一棵 n 个点的无根树，树边的长度都是 1。

对于树上的每个点 i，可以向距离 i 不超过 val_i 的所有点发送命令。

国王的命令

给定一棵 n 个点的无根树，树边的长度都是 1。

对于树上的每个点 i，可以向距离 i 不超过 val_i 的所有点发送命令。

求从 1 号点发送命令到每个点所需的最少转发次数。

国王的命令

给定一棵 n 个点的无根树，树边的长度都是 1。

对于树上的每个点 i，可以向距离 i 不超过 val_i 的所有点发送命令。

求从 1 号点发送命令到每个点所需的最少转发次数。

- $2 \leq n \leq 100000$。

国王的命令

给定一棵 n 个点的无根树，树边的长度都是 1。

对于树上的每个点 i，可以向距离 i 不超过 val_i 的所有点发送命令。

求从 1 号点发送命令到每个点所需的最少转发次数。

- $2 \leq n \leq 100000$。
- Source： 2017 年浙江理工大学程序设计竞赛校赛

Solution

- 从 1 号点出发进行 BFS 即可得到 1 到每个点的最短路。

Solution

- 从 1 号点出发进行 BFS 即可得到 1 到每个点的最短路。
- 直接 BFS 是 $O(n^2)$ 的，不能承受。

Solution

- 从 1 号点出发进行 BFS 即可得到 1 到每个点的最短路。
- 直接 BFS 是 $O(n^2)$ 的，不能承受。
- 瓶颈在于寻找尚未访问过的所有距离 x 不超过 val_x 的点。

Solution

- 对树进行点分治，对每个点记录它被哪些重心所管辖。

Solution

- 对树进行点分治，对每个点记录它被哪些重心所管辖。
- 对于每个重心进行一次 BFS，就可以按到重心的距离从小到大依次保存每个点。

Solution

- 对树进行点分治，对每个点记录它被哪些重心所管辖。
- 对于每个重心进行一次 BFS，就可以按到重心的距离从小到大依次保存每个点。
- 对于 x，枚举所有管辖它的重心 y，不断取出 y 保存的点集里距离最近的点，直到距离超过 $val_x - dist(x, y)$。

Solution

- 对树进行点分治，对每个点记录它被哪些重心所管辖。
- 对于每个重心进行一次 BFS，就可以按到重心的距离从小到大依次保存每个点。
- 对于 x，枚举所有管辖它的重心 y，不断取出 y 保存的点集里距离最近的点，直到距离超过 $val_x - dist(x, y)$。
- 每个点只会被 $O(\log n)$ 个重心管辖，故时间复杂度为 $O(n \log n)$。

树上的最短路

给定一棵 n 个点的无根树，树边的长度为 t_i。

树上的最短路

给定一棵 n 个点的无根树，树边的长度为 t_i。

有 m 条下水道，每条信息为：对于树上 l_1 和 r_1 最短路径上的任意节点 x，l_2 和 r_2 最短路径上的任意节点 y，x 都可以通过 c_i 的代价走到 y。

树上的最短路

给定一棵 n 个点的无根树，树边的长度为 t_i。

有 m 条下水道，每条信息为：对于树上 l_1 和 r_1 最短路径上的任意节点 x，l_2 和 r_2 最短路径上的任意节点 y，x 都可以通过 c_i 的代价走到 y。

求每个点到达目标节点 K 的最少代价。

树上的最短路

给定一棵 n 个点的无根树，树边的长度为 t_i。

有 m 条下水道，每条信息为：对于树上 l_1 和 r_1 最短路径上的任意节点 x，l_2 和 r_2 最短路径上的任意节点 y，x 都可以通过 c_i 的代价走到 y。

求每个点到达目标节点 K 的最少代价。

- $1 \leq n \leq 250000$。

树上的最短路

给定一棵 n 个点的无根树，树边的长度为 t_i。

有 m 条下水道，每条信息为：对于树上 l_1 和 r_1 最短路径上的任意节点 x，l_2 和 r_2 最短路径上的任意节点 y，x 都可以通过 c_i 的代价走到 y。

求每个点到达目标节点 K 的最少代价。

- $1 \leq n \leq 250000$。
- $0 \leq m \leq 100000$。

树上的最短路

给定一棵 n 个点的无根树，树边的长度为 t_i。

有 m 条下水道，每条信息为：对于树上 l_1 和 r_1 最短路径上的任意节点 x，l_2 和 r_2 最短路径上的任意节点 y，x 都可以通过 c_i 的代价走到 y。

求每个点到达目标节点 K 的最少代价。

- $1 \leq n \leq 250000$。
- $0 \leq m \leq 100000$。
- Source：BZOJ 4699

Solution

- 树链剖分 + 线段树优化建图？

Solution

- 树链剖分 + 线段树优化建图？
- 点数 $O(n+m)$，边数 $O(m\log^2 n)$，TLE+MLE。

Solution

- 树链剖分 + 线段树优化建图？
- 点数 $O(n+m)$，边数 $O(m\log^2 n)$，TLE+MLE。
- 倍增优化建图？

Solution

- 树链剖分 + 线段树优化建图？
- 点数 $O(n+m)$，边数 $O(m\log^2 n)$，TLE+MLE。
- 倍增优化建图？
- 点数 $O(n\log n)$，边数 $O(n\log n+m)$，TLE+MLE。

Solution

- 树链剖分 + 线段树优化建图？
- 点数 $O(n+m)$，边数 $O(m\log^2 n)$，TLE+MLE。
- 倍增优化建图？
- 点数 $O(n\log n)$，边数 $O(n\log n+m)$，TLE+MLE。
- 就算不会 MLE，时间复杂度也至少是 $O(n\log^2 n)$ 级别，不能承受。

Solution

- 树链剖分 + 线段树优化建图？
- 点数 $O(n+m)$，边数 $O(m\log^2 n)$，TLE+MLE。
- 倍增优化建图？
- 点数 $O(n\log n)$，边数 $O(n\log n+m)$，TLE+MLE。
- 就算不会 MLE，时间复杂度也至少是 $O(n\log^2 n)$ 级别，不能承受。
- 利用 Dijkstra 算法本身的特性就可以做到 $O((n+m)\log n)$ 的复杂度。

Solution

- 给定一个点 x，如何取出所有经过它的下水道？

Solution

- 给定一个点 x，如何取出所有经过它的下水道？
- 一条下水道经过 x 等价于它起点在 x 的子树里面且终点不在 x 的子树里面，或者两端点的 LCA 就是 x。

Solution

- 给定一个点 x，如何取出所有经过它的下水道？
- 一条下水道经过 x 等价于它起点在 x 的子树里面且终点不在 x 的子树里面，或者两端点的 LCA 就是 x。
- 对于第一种情况，也就是说起点在 x 的 DFS 序子区间里，终点小于 $st[x]$ 或者大于 $en[x]$。

Solution

- 给定一个点 x，如何取出所有经过它的下水道？
- 一条下水道经过 x 等价于它起点在 x 的子树里面且终点不在 x 的子树里面，或者两端点的 LCA 就是 x。
- 对于第一种情况，也就是说起点在 x 的 DFS 序子区间里，终点小于 $st[x]$ 或者大于 $en[x]$。
- 假设下水道是 $A - B$ 向 $C - D$ 连边，并且 $st[A] < st[B]$。

Solution

- 给定一个点 x，如何取出所有经过它的下水道？
- 一条下水道经过 x 等价于它起点在 x 的子树里面且终点不在 x 的子树里面，或者两端点的 LCA 就是 x。
- 对于第一种情况，也就是说起点在 x 的 DFS 序子区间里，终点小于 $st[x]$ 或者大于 $en[x]$。
- 假设下水道是 $A - B$ 向 $C - D$ 连边，并且 $st[A] < st[B]$。
- 那么只需要不断查询 $st[A]$ 在 $[L, R]$ 里 $st[B]$ 最大的下水道，以及 $st[B]$ 在 $[L, R]$ 里 $st[A]$ 最小的下水道即可。

Solution

- 用线段树按 dfs 序维护终点 *dfn* 最小和最大的下水道，然后不断取出即可。

Solution

- 用线段树按 dfs 序维护终点 dfn 最小和最大的下水道，然后不断取出即可。
- 注意到 Dijkstra 的时候，dis 不降，所以对于一条下水道，只有 $A - B$ 里第一个被访问到的点才会用到它，所以在使用完毕后直接删除即可。

Solution

- 用线段树按 dfs 序维护终点 *dfn* 最小和最大的下水道，然后不断取出即可。
- 注意到 Dijkstra 的时候，*dis* 不降，所以对于一条下水道，只有 $A - B$ 里第一个被访问到的点才会用到它，所以在使用完毕后直接删除即可。
- 如何求最短路？

Solution

- 用线段树按 dfs 序维护终点 dfn 最小和最大的下水道，然后不断取出即可。
- 注意到 Dijkstra 的时候，dis 不降，所以对于一条下水道，只有 $A - B$ 里第一个被访问到的点才会用到它，所以在使用完毕后直接删除即可。
- 如何求最短路？
- 考虑转化问题，改成求起点到每条有向边的最短路，这个最短路包括这条边本身的权值。

Solution

- 按距离维护一个小根堆，里面有两种元素：(1) 某条有向树边 $x \to y$。(2) 某条下水道 $A - B, C - D$。

Solution

- 按距离维护一个小根堆，里面有两种元素：(1) 某条有向树边 $x \to y$。(2) 某条下水道 $A - B, C - D$。
- 每次取出堆顶元素，如果是树边，那么枚举 y 的出边，同时将经过 y 的下水道都取出即可。

Solution

- 按距离维护一个小根堆，里面有两种元素：(1) 某条有向树边 $x \to y$。(2) 某条下水道 $A - B, C - D$。
- 每次取出堆顶元素，如果是树边，那么枚举 y 的出边，同时将经过 y 的下水道都取出即可。
- 如果是下水道，那么暴力将 $C - D$ 路径上所有没有访问过的点的出边都取出。

Solution

- 按距离维护一个小根堆，里面有两种元素：(1) 某条有向树边 $x \to y$。(2) 某条下水道 $A - B, C - D$。
- 每次取出堆顶元素，如果是树边，那么枚举 y 的出边，同时将经过 y 的下水道都取出即可。
- 如果是下水道，那么暴力将 $C - D$ 路径上所有没有访问过的点的出边都取出。
- 每个点只需要取一次，用并查集完成路径压缩即可。

Solution

- 按距离维护一个小根堆，里面有两种元素：(1) 某条有向树边 $x \to y$。(2) 某条下水道 $A - B, C - D$。
- 每次取出堆顶元素，如果是树边，那么枚举 y 的出边，同时将经过 y 的下水道都取出即可。
- 如果是下水道，那么暴力将 $C - D$ 路径上所有没有访问过的点的出边都取出。
- 每个点只需要取一次，用并查集完成路径压缩即可。
- 时间复杂度 $O((n+m)\log n)$，空间复杂度 $O(n+m)$。

Summary

- 利用 Dijkstra 本身的特性就可以优化最短路的复杂度。

Summary

- 利用 Dijkstra 本身的特性就可以优化最短路的复杂度。
- 以上问题均是明确的最短路模型，考察建图或者求最短路的优化。

Summary

- 利用 Dijkstra 本身的特性就可以优化最短路的复杂度。
- 以上问题均是明确的最短路模型，考察建图或者求最短路的优化。
- 其它一些问题则是隐式的最短路，比较难以看出模型。

狼抓兔子

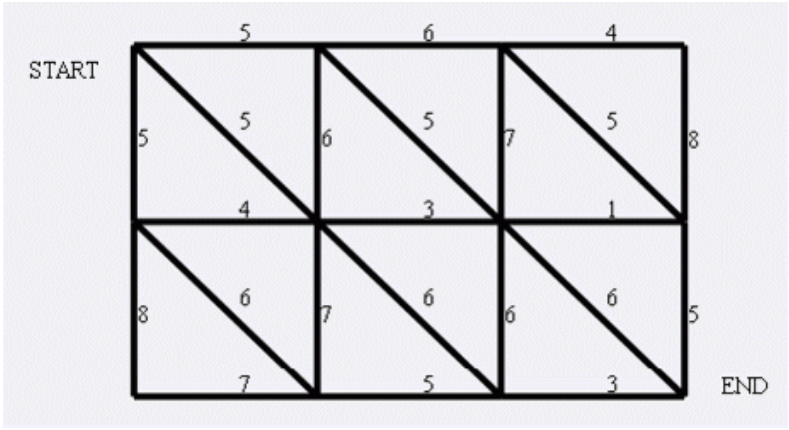
给定一张 $n \times m$ 的网格图，左上角为源点，右下角为汇点。

狼抓兔子

给定一张 $n \times m$ 的网格图，左上角为源点，右下角为汇点。求这个图的最小割。

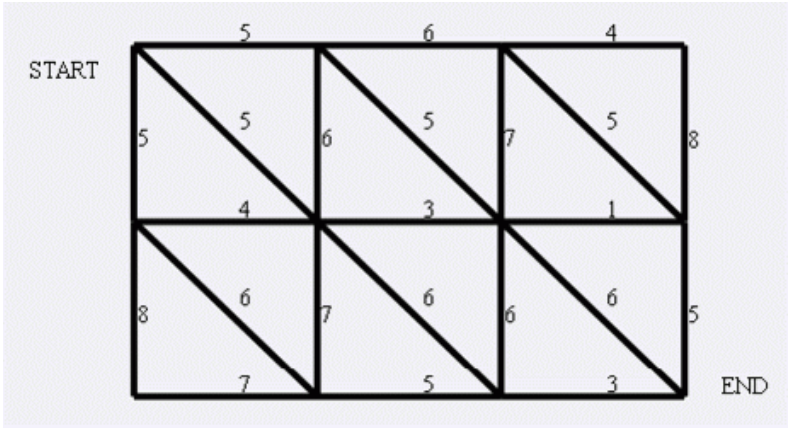
狼抓兔子

给定一张 $n \times m$ 的网格图，左上角为源点，右下角为汇点。求这个图的最小割。



狼抓兔子

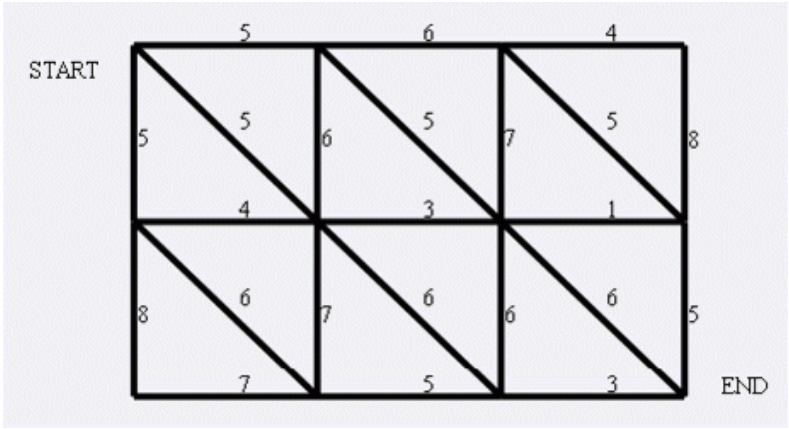
给定一张 $n \times m$ 的网格图，左上角为源点，右下角为汇点。求这个图的最小割。



- $1 \leq n, m \leq 1000$。

狼抓兔子

给定一张 $n \times m$ 的网格图，左上角为源点，右下角为汇点。求这个图的最小割。



- $1 \leq n, m \leq 1000$。
- Source： BZOJ 1001

Solution

- 将每个三角形看成一个区域，右上角空白与左下角空白也看成区域。

Solution

- 将每个三角形看成一个区域，右上角空白与左下角空白也看成区域。
- 若左上角与右下角不连通，那么右上角与左下角通过割边以及一些三角形区域连通。

Solution

- 将每个三角形看成一个区域，右上角空白与左下角空白也看成区域。
- 若左上角与右下角不连通，那么右上角与左下角通过割边以及一些三角形区域连通。
- 相邻区域之间连边，求右上角与左下角的最短路即可。

Solution

- 将每个三角形看成一个区域，右上角空白与左下角空白也看成区域。
- 若左上角与右下角不连通，那么右上角与左下角通过割边以及一些三角形区域连通。
- 相邻区域之间连边，求右上角与左下角的最短路即可。
- 时间复杂度 $O(nm\log(nm))$。

Treasures and Vikings

给定一张 $n \times m$ 的网格图，每个点是障碍或海洋。

Treasures and Vikings

给定一张 $n \times m$ 的网格图，每个点是障碍或海洋。

上面有 3 个点，分别表示你、海盗以及宝藏的位置。

Treasures and Vikings

给定一张 $n \times m$ 的网格图，每个点是障碍或海洋。

上面有 3 个点，分别表示你、海盗以及宝藏的位置。

每一回合你可以向上下左右移动一步，然后海盗选择向上下左右移动一步，也可以不动。

Treasures and Vikings

给定一张 $n \times m$ 的网格图，每个点是障碍或海洋。

上面有 3 个点，分别表示你、海盗以及宝藏的位置。

每一回合你可以向上下左右移动一步，然后海盗选择向上下左右移动一步，也可以不动。

回合结束时，若你与海盗在同一行或同一列，且中间没有障碍遮挡，那么会被杀死，否则若你碰到了宝藏，则胜利。

Treasures and Vikings

给定一张 $n \times m$ 的网格图，每个点是障碍或海洋。

上面有 3 个点，分别表示你、海盗以及宝藏的位置。

每一回合你可以向上下左右移动一步，然后海盗选择向上下左右移动一步，也可以不动。

回合结束时，若你与海盗在同一行或同一列，且中间没有障碍遮挡，那么会被杀死，否则若你碰到了宝藏，则胜利。

请判断是否存在一条路线，使得海盗不管怎么走你都可以拿到宝藏。

Treasures and Vikings

给定一张 $n \times m$ 的网格图，每个点是障碍或海洋。

上面有 3 个点，分别表示你、海盗以及宝藏的位置。

每一回合你可以向上下左右移动一步，然后海盗选择向上下左右移动一步，也可以不动。

回合结束时，若你与海盗在同一行或同一列，且中间没有障碍遮挡，那么会被杀死，否则若你碰到了宝藏，则胜利。

请判断是否存在一条路线，使得海盗不管怎么走你都可以拿到宝藏。

- $2 \leq n, m \leq 700$。

Treasures and Vikings

给定一张 $n \times m$ 的网格图，每个点是障碍或海洋。

上面有 3 个点，分别表示你、海盗以及宝藏的位置。

每一回合你可以向上下左右移动一步，然后海盗选择向上下左右移动一步，也可以不动。

回合结束时，若你与海盗在同一行或同一列，且中间没有障碍遮挡，那么会被杀死，否则若你碰到了宝藏，则胜利。

请判断是否存在一条路线，使得海盗不管怎么走你都可以拿到宝藏。

- $2 \leq n, m \leq 700$。
- Source : Baltic Olympiad In Informatics 2011

Solution

- 海盗的最优策略是让每个点尽早被自己守住。

Solution

- 海盗的最优策略是让每个点尽早被自己守住。
- BFS 求出海盗到每个点的最短路，再通过四遍递推求出每个点最早被海盗看见的时间。

Solution

- 海盗的最优策略是让每个点尽早被自己守住。
- BFS 求出海盗到每个点的最短路，再通过四遍递推求出每个点最早被海盗看见的时间。
- 用起点开始 BFS，若能早于海盗到达某个点，才能判定为可达。

Solution

- 海盗的最优策略是让每个点尽早被自己守住。
- BFS 求出海盗到每个点的最短路，再通过四遍递推求出每个点最早被海盗看见的时间。
- 用起点开始 BFS，若能早于海盗到达某个点，才能判定为可达。
- 时间复杂度 $O(nm)$。

Sums

给定正整数 $a_1, a_2, ..., a_n$ 以及 m 个询问，每次给出一个 k，试判断方程 $a_1x_1 + a_2x_2 + ... + a_nx_n = k$ 是否存在非负整数解。

Sums

给定正整数 $a_1, a_2, ..., a_n$ 以及 m 个询问，每次给出一个 k，试判断方程 $a_1x_1 + a_2x_2 + ... + a_nx_n = k$ 是否存在非负整数解。

- $1 \leq n \leq 5000$。

Sums

给定正整数 $a_1, a_2, ..., a_n$ 以及 m 个询问，每次给出一个 k，试判断方程 $a_1x_1 + a_2x_2 + ... + a_nx_n = k$ 是否存在非负整数解。

- $1 \leq n \leq 5000$。
- $1 \leq m \leq 10000$。

Sums

给定正整数 $a_1, a_2, ..., a_n$ 以及 m 个询问，每次给出一个 k，试判断方程 $a_1x_1 + a_2x_2 + ... + a_nx_n = k$ 是否存在非负整数解。

- $1 \leq n \leq 5000$。
- $1 \leq m \leq 10000$。
- $1 \leq a_i \leq 50000$。

Sums

给定正整数 $a_1, a_2, ..., a_n$ 以及 m 个询问，每次给出一个 k，试判断方程 $a_1x_1 + a_2x_2 + ... + a_nx_n = k$ 是否存在非负整数解。

- $1 \leq n \leq 5000$。
- $1 \leq m \leq 10000$。
- $1 \leq a_i \leq 50000$。
- $0 \leq k \leq 10^9$。

Sums

给定正整数 $a_1, a_2, ..., a_n$ 以及 m 个询问，每次给出一个 k，试判断方程 $a_1x_1 + a_2x_2 + ... + a_nx_n = k$ 是否存在非负整数解。

- $1 \leq n \leq 5000$。
- $1 \leq m \leq 10000$。
- $1 \leq a_i \leq 50000$。
- $0 \leq k \leq 10^9$。
- Source：POI 2003

Solution

- 不妨设 a_1 是序列中最小的。

Solution

- 不妨设 a_1 是序列中最小的。
- 若方程 $a_1x_1 + a_2x_2 + \ldots + a_nx_n = k$ 存在非负整数解，那么 $k + a_1$ 也必然有解。

Solution

- 不妨设 a_1 是序列中最小的。
- 若方程 $a_1x_1 + a_2x_2 + \ldots + a_nx_n = k$ 存在非负整数解，那么 $k + a_1$ 也必然有解。
- 建立 a_1 个点的图，点编号依次为 0 到 $a_1 - 1$，i 向 $(i + a_j) \bmod a_1$ 连边，边权 a_j。

Solution

- 不妨设 a_1 是序列中最小的。
- 若方程 $a_1x_1 + a_2x_2 + \ldots + a_nx_n = k$ 存在非负整数解，那么 $k + a_1$ 也必然有解。
- 建立 a_1 个点的图，点编号依次为 0 到 $a_1 - 1$，i 向 $(i + a_j) \bmod a_1$ 连边，边权 a_j。
- 求出 0 到每个点的最短路 dis_i，即表示最小的有解的 k，且 k 满足 $k \bmod a_1 = i$。

Solution

- 不妨设 a_1 是序列中最小的。
- 若方程 $a_1x_1 + a_2x_2 + \ldots + a_nx_n = k$ 存在非负整数解，那么 $k + a_1$ 也必然有解。
- 建立 a_1 个点的图，点编号依次为 0 到 $a_1 - 1$，i 向 $(i + a_j) \bmod a_1$ 连边，边权 a_j。
- 求出 0 到每个点的最短路 dis_i，即表示最小的有解的 k，且 k 满足 $k \bmod a_1 = i$。
- 那么若 $dis_{k \bmod a_1} \leq k$，则 k 有解。

Solution

- 不妨设 a_1 是序列中最小的。
- 若方程 $a_1x_1 + a_2x_2 + \ldots + a_nx_n = k$ 存在非负整数解，那么 $k + a_1$ 也必然有解。
- 建立 a_1 个点的图，点编号依次为 0 到 $a_1 - 1$，i 向 $(i + a_j) \bmod a_1$ 连边，边权 a_j。
- 求出 0 到每个点的最短路 dis_i，即表示最小的有解的 k，且 k 满足 $k \bmod a_1 = i$。
- 那么若 $dis_{k \bmod a_1} \leq k$，则 k 有解。
- 时间复杂度 $O(na_1 \log(na_1) + m)$。

LongLongTripDiv1

给定一张 n 个点 m 条边的无向带权图，问是否存在一条从 1 号点到 n 号的长度为 T 的路径，不需要是简单路径。

LongLongTripDiv1

给定一张 n 个点 m 条边的无向带权图，问是否存在一条从 1 号点到 n 号的长度为 T 的路径，不需要是简单路径。

- $2 \leq n \leq 50$。

LongLongTripDiv1

给定一张 n 个点 m 条边的无向带权图，问是否存在一条从 1 号点到 n 号的长度为 T 的路径，不需要是简单路径。

- $2 \leq n \leq 50$。
- $1 \leq m \leq 50$。

LongLongTripDiv1

给定一张 n 个点 m 条边的无向带权图，问是否存在一条从 1 号点到 n 号的长度为 T 的路径，不需要是简单路径。

- $2 \leq n \leq 50$。
- $1 \leq m \leq 50$。
- $1 \leq T \leq 10^9$。

LongLongTripDiv1

给定一张 n 个点 m 条边的无向带权图，问是否存在一条从 1 号点到 n 号的长度为 T 的路径，不需要是简单路径。

- $2 \leq n \leq 50$。
- $1 \leq m \leq 50$。
- $1 \leq T \leq 10^9$。
- $1 \leq w_i \leq 10000$。

LongLongTripDiv1

给定一张 n 个点 m 条边的无向带权图，问是否存在一条从 1 号点到 n 号的长度为 T 的路径，不需要是简单路径。

- $2 \leq n \leq 50$。
- $1 \leq m \leq 50$。
- $1 \leq T \leq 10^9$。
- $1 \leq w_i \leq 10000$。
- Source：SRM 615

Solution

- 取一条与 1 相连的权值最小的边 w。

Solution

- 取一条与 1 相连的权值最小的边 w。
- 若存在一条从 1 到 n 的长度为 k 的路径，那么必然存在一条长度为 $k+2w$ 的路径，只要一开始在那条边上往返走就好了。

Solution

- 取一条与 1 相连的权值最小的边 w。
- 若存在一条从 1 到 n 的长度为 k 的路径，那么必然存在一条长度为 $k+2w$ 的路径，只要一开始在那条边上往返走就好了。
- 设 $f_{i,j}$ 表示从 1 到 i，路径长度模 $2w$ 为 j 时，路径长度的最小值。

Solution

- 取一条与 1 相连的权值最小的边 w。
- 若存在一条从 1 到 n 的长度为 k 的路径，那么必然存在一条长度为 $k+2w$ 的路径，只要一开始在那条边上往返走就好了。
- 设 $f_{i,j}$ 表示从 1 到 i，路径长度模 $2w$ 为 j 时，路径长度的最小值。
- 用最短路算法求出所有 $f_{i,j}$，然后检查 $f_{n,T \bmod 2w}$ 是否不超过 T 即可。

Solution

- 取一条与 1 相连的权值最小的边 w。
- 若存在一条从 1 到 n 的长度为 k 的路径，那么必然存在一条长度为 $k+2w$ 的路径，只要一开始在那条边上往返走就好了。
- 设 $f_{i,j}$ 表示从 1 到 i，路径长度模 $2w$ 为 j 时，路径长度的最小值。
- 用最短路算法求出所有 $f_{i,j}$，然后检查 $f_{n,T \bmod 2w}$ 是否不超过 T 即可。
- 时间复杂度 $O(nw\log(nw))$。

课后习题

- Meeting (HDU 5521)

课后习题

- Meeting (HDU 5521)
- The Captain (AMPPZ 2014)

课后习题

- Meeting (HDU 5521)
- The Captain (AMPPZ 2014)
- Tax (PA2012 Final)

课后习题

- Meeting (HDU 5521)
- The Captain (AMPPZ 2014)
- Tax (PA2012 Final)
- In Touch (HDU 5361)

课后习题

- Meeting (HDU 5521)
- The Captain (AMPPZ 2014)
- Tax (PA2012 Final)
- In Touch (HDU 5361)
- Road (HDU 5669)

课后习题

- Meeting (HDU 5521)
- The Captain (AMPPZ 2014)
- Tax (PA2012 Final)
- In Touch (HDU 5361)
- Road (HDU 5669)
- 飞飞侠 (BZOJ 2143)

课后习题

- Meeting (HDU 5521)
- The Captain (AMPPZ 2014)
- Tax (PA2012 Final)
- In Touch (HDU 5361)
- Road (HDU 5669)
- 飞飞侠 (BZOJ 2143)
- Advanced Traffic System (HDU 5910)

课后习题

- Meeting (HDU 5521)
- The Captain (AMPPZ 2014)
- Tax (PA2012 Final)
- In Touch (HDU 5361)
- Road (HDU 5669)
- 飞飞侠 (BZOJ 2143)
- Advanced Traffic System (HDU 5910)
- [Noi2010] 海拔 (BZOJ 2007)

Thank you!