# 单源最短路径

Dijkskra算法是基于贪心算法的。松弛（relax）

有负环路图存在：最短路径不存在。可以循环使花费越来越低，没有最低

Bellman-ford：找到所有最短路径的长度从一个源sV到vV或者……………

时间复杂度为：E+Vlogv

Bellman\_ford算法：<https://blog.csdn.net/sunnyyoona/article/details/45222073>

松弛：原来用一根橡皮筋连接p和w两点，现在有一点v到w的路径更短，现在把橡皮筋w点的另一端p换成v点，这样缓解橡皮筋紧绷的压力，让其变得松弛。

1. 初始化，定义d[i]为从源点到其他各点的最短路径，d[0]=0;d[i]=正无穷

d[s]🡨0;

for each v ∈V-{s}

d[s]🡨∞.

1. 迭代求解：反复对边集E中的每条边进行松弛操作，使得顶点集V中的每个顶点v的最短距离估计值逐步逼近其最短距离（运行|V|-1次）。
2. 检验负权回路：判断边集E中的每一条边的两个端点是否收敛。如果存在未收敛的顶点，则算法返回false，表明问题无解；否则算法返回true，并且从源点可达的顶点v的最短距离保存在 dist[v]中。
3. 时间复杂度为（VE）

# All-pairs shortest paths：求图中任意两点的最短距离

可以把Dijkstra算法运行v遍，时间复杂度VE+V^2lgV

如果图中有负权值在，就不能用Dijkstra算法。