# Dijkstra正确性证明

Updated 1702 GMT+8 Dec 02 2024

2024 fall, Complied by Hongfei Yan

Proof of Dijkstra's Correctness

# 1 详细解释

Dijkstra算法的正确性证明主要基于贪心选择性质和最优子结构性质。下面是对Dijkstra算法正确性的详细解释：

\*\*贪心选择性质\*\*

Dijkstra算法在每一步中总是选择当前已知最短路径的顶点，并且更新其邻居顶点的距离。这种选择方式确保了每次添加到最终解中的顶点都是当前最优的选择。

\*\*最优子结构\*\*

如果从起点 `s` 到某个顶点 `v` 的最短路径是通过顶点 `u`，那么从 `s` 到 `u` 的部分也必须是最短路径。这保证了局部最优解可以组合成全局最优解。

\*\*证明步骤\*\*

1. \*\*定义\*\*：

- 让 `S` 表示已经确定了最短路径的顶点集合。

- 让 `V-S` 表示尚未确定最短路径的顶点集合。

- `d[v]` 表示从起点 `s` 到顶点 `v` 的当前已知最短距离。

- $\delta(s, v) $ 表示从起点 `s` 到顶点 `v` 的实际最短距离。

2. \*\*初始状态\*\*：

- 算法开始时，$S = \{s\}$ ，即只包含起点 `s`。

- 对于所有顶点 $ v \in V-S $，初始化 `d[v]` 为从 `s` 到 `v` 的直接边的权重（如果存在），否则为无穷大。

3. \*\*不变量\*\*：

- 在每一步执行之前，对于所有 $ u \in S $，有 $ d[u] = \delta(s, u) $。

- 对于所有 $ v \in V-S $，有 $ d[v] \geq \delta(s, v) $。

4. \*\*迭代过程\*\*：

- 在每一步中，选择 `V-S` 中 `d[v]` 最小的顶点 `u` 加入 `S`。

- 更新 `u` 的所有邻居 `v` 的 `d[v]` 值，如果通过 `u` 到达 `v` 的新路径更短，则更新 `d[v]`。

5. \*\*Dijkstra正确性证明，如何理解？\*\*：

- 假设在某一步骤中，我们选择了 `u` 加入 `S` ，并且 $ u \neq s $。

- 由于 `u` 是 `V-S` 中 `d` 值最小的顶点，因此 $ d[u] \leq d[v] $ 对于所有 $ v \in V-S $ 成立。

- 根据不变量，$ d[u] \geq \delta(s, u) $。

- 如果 $ d[u] > \delta(s, u) $，则存在一条从 `s` 到 `u` 的更短路径，但这条路径必须经过 `V-S` 中的某个顶点 `w`（因为 `u` 是第一个被加入 S 的顶点）。

- 由于$ d[w] \geq \delta(s, w) $，且 $ \delta(s, w) + \text{weight}(w, u) \geq \delta(s, u) $，所以 $ d[u] $ 不可能大于 $ \delta(s, u) $。

> 由于我们假设了存在一条更短的路径，即d[u] > δ(s, u)，那么按照Dijkstra算法更新规则，d[u]应该被更新为d[w] + weight(w, u)或更小的值。这与d[u] > δ(s, u)相矛盾，因为这样会导致d[u]不大于δ(s, u)。

- 因此，$ d[u] = \delta(s, u) $。

6. \*\*终止条件\*\*：

- 当所有顶点都被加入 `S` 时，算法结束。

- 此时，对于所有顶点 `v` ，$ d[v] = \delta(s, v) $。

\*\*结论\*\*

通过上述证明，我们可以得出结论：Dijkstra算法能够正确地找到从单个源点到图中所有其他顶点的最短路径。该算法依赖于非负权重边的假设，如果图中存在负权重边，Dijkstra算法可能会给出错误的结果。在这种情况下，可以使用Bellman-Ford算法来处理。

# 2 进一步解释

Dijkstra 算法的正确性证明基于以下核心逻辑：\*\*每次将一个顶点 `u` 加入已确定最短路径集合 `S` 时，`d[u]` 必然等于从起点 `s` 到该顶点 `u` 的真实最短路径权值 $\delta(s, u)$\*\*。以下是如何理解这一证明步骤的关键点：

------

\*\*1. `u` 的选择保证了它的最小性\*\*

- 在算法中，每次选择 `u` 时，其 `d[u]` 是所有 `V-S` 中 `d` 值最小的。

- 换句话说，在尚未被处理的顶点中，`u` 是当前最接近起点 `s` 的顶点。

因此，$d[u] \leq d[v]$ 对于所有 $v \in V-S$。

\*\*2. 不变量：$d[u] \geq \delta(s, u)$\*\*

- 算法的初始化确保了对所有顶点 $v$，`d[v]` 是从起点 `s` 出发到达该顶点的最短路径的一个上界（初始化时，$d[s]=0$，其余顶点 $d[v]=\infty$）。

- 在算法每一步中，通过松弛操作不断缩小 `d[v]` 的值，但始终保持 $d[v] \geq \delta(s, v)$。

\*\*3. 假设反证法：如果 $d[u] > \delta(s, u)$\*\*

如果 $d[u] > \delta(s, u)$，意味着存在更短的路径从 `s` 到达 `u`。设这条路径为 $s \to w \to u$，其中 $w \in V-S$ 是路径上未处理的某个顶点。

\*\*矛盾点分析\*\*

- 根据不变量，$d[w] \geq \delta(s, w)$。

- 由于 `u` 是当前 `V-S` 中 `d` 最小的顶点，因此 $d[u] \leq d[w]$。

- 另一方面，路径 $s \to w \to u$ 的真实距离为 $\delta(s, w) + \text{weight}(w, u)$，而 $\delta(s, w) + \text{weight}(w, u) \geq \delta(s, u)$。

> 由于我们假设了存在一条更短的路径，即 d[u] > δ(s, u)，那么按照Dijkstra算法更新规则，d[u]应该被更新为d[w] + weight(w, u)或更小的值。这与d[u] > δ(s, u)相矛盾，因为这样会导致d[u]不大于δ(s, u)。

- 综合以上推导可知，$d[u] \geq \delta(s, u)$。

但 $d[u] > \delta(s, u)$ 的假设与上述结论矛盾。

\*\*4. 结论：$d[u] = \delta(s, u)$\*\*

由于不存在更短路径未被考虑，因此 `d[u]` 必等于从 `s` 到 `u` 的真实最短路径权值 $\delta(s, u)$。

# 3 直观理解

可以将 Dijkstra 算法看作“逐步揭露最短路径”的过程：

1. 每次处理一个顶点 `u`，它已经是离 `s` 最近的、尚未处理的顶点。

2. 对于 `u`，我们确认其最短路径值为 $d[u] = \delta(s, u)$，并将其固定在 `S` 中。

3. 此后更新其邻接顶点的 `d` 值，使得其他顶点的潜在路径长度不断逼近真实最短路径。

这种逐步扩展的方式确保了算法的正确性。