Floyd 算法正确性的形式化验证

刘衍

2025 年 12 月 6 日

1 算法介绍

Floyd 算法是一个动态规划算法，用于计算图中所有顶点对之间的最短路径。它的核心思想是，通过逐步允许使用更多的顶点作为中间点，来不断优化路径。Floyd 算法可以正确处理有向图和负权（但不可存在负权回路）的最短路径问题，同时也被用于计算有向图的传递闭包。

在这里，我们主要证明对简单有向的有限非负权图进行 Floyd 算法得到的距离方阵正确描述了任意两个点之间最短路径的关系。

2 算法描述

Algorithm 1 Floyd Algorithm

Require: Undirected, Simple, Nonnegative-Weighted Graph $G(V, E)$, *dist* is a $|V|\times|V|$ array of minimum distances initialized to $+\infty$

Ensure: $dist[i][j]$ discribes the length of the shortest path from vertex i to vertex j.

```
for each edge (u, v) do
    dist[u][v] = w(u, v)
end for
for each vertex v do
    dist[v][v] = 0
end for                          ▷ Can also be wrapped into initial state or the first step of loop.
for k from 1 to |V| do
    for i from 1 to |V| do
        for j from 1 to |V| do
            if dist[i][j] > dist[i][k] + dist[k][j] then
                dist[i][j] = dist[i][k] + dist[k][j]
            end if
        end for
    end for
end for
```

考虑一个图 G，其顶点 V 编号为 1 到 N。进一步考虑一个函数 $shortestPath(i, j, k)$，它返回从 i 到 j 的最短可能路径长度（如果存在），且路径中仅使用集合 $\{1, 2, \ldots, k\}$ 中的顶点作为中间点。现在，给定这个函数，我们的目标是找到从每个 i 到每个 j 的最短路径长度，路径中可以使用 $\{1, 2, \ldots, N\}$ 中的任何顶点。根据定义，这就是值 $shortestPath(i, j, N)$，我们将递归地计算出它。

注意到 $shortestPath(i, j, k) \leq shortedtPath(i, j, k-1)$，因为后者的路径都包含在前者中。

(1) 如果仅使用集合 $\{1, 2, \ldots, k\}$ 中的顶点的从 i 到 j 的最短路径不经过 k，则这条路径仅使用集合 $\{1, 2, \ldots, k-1\}$ 中的顶点从 i 到达了 j，意味着 $shortestPath(i, j, k) = shortedtPath(i, j, k-1)$。

(2) 如果仅使用集合 $\{1, 2, \ldots, k\}$ 中的顶点的从 i 到 j 的最短路径经过 k，由于没有负权回路，我们可以将这条路径分成两段：
一条从 i 到 k 的路径，该路径使用顶点 $\{1, 2, \ldots, k-1\}$
一条从 k 到 j 的路径，该路径使用顶点 $\{1, 2, \ldots, k-1\}$
最短路径上的任意路径也都是最短路径。所以也就得到了
$shortestPath(i, j, k) = shortedtPath(i, k, k-1) + shortestPath(k, j, k-1)$

3 任务指导

我们给出了 Monad 算子定义的 Floyd 算法程序，见 Floyd.v，并且我们给出了关于 (最短) 路径的性质列表。
(A) 证明性质列表中的性质。
(B) 证明 Floyd.v 中关于 Floyd 算法的正确性 (Theorem *Floyd_correct*) 的描述。你需要在合适的位置写出循环不变量，并使用性质列表中的性质完成不变量的证明。
例如：$k-$ 循环的一个循环不变量是：$dist[i][j] = shortestPath(i, j, k)$;
$i-$ 循环的几个可能的不变量是：
$\forall i_0 (< i) \forall j, dist[i][j] = shortestPath(i_0, j, k)$
$\forall i_0 (\geq i) \forall j, dist[i][j] = shortestPath(i_0, j, k-1)$。

4 性质列表

性质的形式化描述详见 floyd.v。

Lemma (path_splice_trans). 从 i 到 k 的路径拼接上从 k 到 j 的路径是一条从 i 到 j 的路径。

Lemma (path_splice_length). 从 i 到 k 的路径拼接上从 k 到 j 的路径的长度等于这两条路径的长度之和。

Lemma (path_in_vset_mono). 从 i 到 j 限制在 *1,2,...,k-1* 上的路径也是限制在 *1,2,...,k* 上的路径。

Lemma (shortest_path_all_shortest). 从 i 到 j 的最短路径的任何两个顶点在这条路径中的路径都是一条最短路径。

Lemma (shortest_path_is_simple). 任意的最短路径都是简单路径。

Lemma (shortest_path_segment). 从 i 到 j 限制在 *1,2,...,k* 的最短路径如果经过 k, 则可以把路径切分为从 i 到 k 和从 k 到 j 限制在 *1,2,...,k-1* 上的两条最短路径。

Lemma (shortest_path_triangle). 从 i 到 j 的最短路径长度不大于从 i 到 k 的最短路径长度加上从 k 到 j 的最短路径长度。