Kruskal 算法正确性的形式化验证

刘衍

2025 年 12 月 6 日

1 算法介绍

Kruskal 算法是一个寻找最小生成森林的边贪心算法。如果图是连通的，它将找到一个最小生成树。它按权重升序排列的顺序遍历图的边，添加加入这条边到已有边集中时候而不会形成环的边，来构建这棵树。

在这里。我们主要证明对一个简单无向连通的非负权图，对它进行 kruskal 算法得到的边集能够构成一个原图的最小生成树。

2 算法描述

Algorithm 1 kruskal Algorithm

Require: Undirected, Simple, Connected, Positive-Weighted Graph $G(V, E)$, two empty edge set *visitedEdges* and *spanningEdges*.
Ensure: *spanningEdges* constructs a MST of G

```
while !(visitedEdges = E) do
    e = (u, v) with minimum weight
    if !(have_circuit (spanningEdges + e)) then
        spanningEdges+ = [e]
    end if
end while
```

我们证明每次循环都不会改变 *spanningEdges* 构成的图是某个最小生成树的子图。换句话说，我们来证明这是一个循环不变量。

假设循环开始时的 spanningEdges 构成的图 Y_1 是原图的某个最小生成树 T_1 的子图，并且选中 $e = (x, y)$ 是权重最小的未访问过的边，来证明把循环结束时 spanningEdges 构成的图 Y_2 也包含于某个最小生成树 T_2 中。

如果 $Y_1 + e$ 会形成环，那么 $Y_2 = Y_1$ 将保持不变，取 $T_2 = T_1$ 即可。下面我们来考虑 $Y_1 + e$ 不会形成环的情形。

(1)$e \in T_1$，则取 $T_2 = T_1$ 即可。

(2)$e \notin T_1$，则由于树的简单性质，$T + e$ 上存在环 c。取环 c 上还未被访问的另一条边 $f \neq e$。

① 权重 $f.weight < e.weight$，与 e 是权重最小的未访问的边矛盾。

② 权重 $f.weight \geq e.weight$，则 $T_1 + e - f$ 的权值不大于 T_1，并且由于从环上删去条边不改变图的连通性，所以 $T_1 + e - f$ 是一颗树。于是我们找到了权值不大于 T_1 的包含 Y_2 的最小生成树 $T_2 = T_1 + e - f$。

3 任务指导

你已经被给出了 Monad 算子定义的 Kruskal 算法程序，见 Kruskal.v，并且我们给出了关于 (最小) 生成树的性质列表。
(A) 证明性质列表中所有关于 (最小) 生成树的性质。
(B) 证明 Kruskal.v 中关于 Kruskal 算法的正确性 (Theorem Kruskal_correct) 的描述。在合适的位置写出需要证明的循环不变量，并使用性质列表中的性质完成不变量的证明。
例如提到的：
当前程序状态得到的图包含于某个原图的最小生成树中。
当前程序状态中的边集不形成环。
已经访问过但不在边集中的边，加入当前程序状态中的边集会形成环。

4 性质列表

性质的形式化描述详见 prim.v。

Lemma (path_simplifier). 如果两个顶点之间存在路径，则这两个顶点之间也存在简单路径。

Lemma (tree_have_path). 树中任意两个顶点之间存在路径。

Lemma (tree_have_unique_simple_path). 树中任意两个顶点之间存在唯一一条简单路径。

Lemma (tree_add_edge_have_circuit). 往树中加入一条边会形成唯一的一条简单回路。

Lemma (remove_edge_on_circuit_preserves_connectivity). 从一个连通图中移除一条属于某个环路的边，图的连通性不受影响。

Lemma (tree_add_edge_delete_on_circuit_is_tree). 往一颗树中加入一条边再删去加入边后形成的简单回路中的任意一条边得到的图仍然是一颗树。

Lemma (list_sum_add_edge). 往树中加入一条边，再删除权重更大的一条边，得到的图的权重和更小。