Shortest path tree proof
for a generic RELAX-basic algorithm

Yefim Dinitz

Preliminaries

Proofs of the shortest paths tree property remained not simple for decades. For example, in both editions of a popular textbook [1], this proof is about three pages long. Other, much shorter proofs appear in [3, 2]. However, they are indirect: the key lemma proves that back-pointers \( \pi \) computed by the algorithm never form a cycle, and this lemma implies the statement.

In contrast, the proof of a similar statement for the Dijkstra algorithm is straightforward: By the algorithm, beginning from the initial tree consisting of root \( s \) only, each iteration adds a new leaf edge to the back-pointer graph; clearly, its property of being a tree rooted at \( s \) is maintained. We suggest a similar proof for the general relaxation-based algorithm, given the correctness of computing the distances.

Recall known properties of Relax-based algorithms:

**Fact 0.1** Consider an arbitrary (properly initialized) sequence of Relax executions.

1. At any moment and for any vertex \( v \), \( d(v) \geq \delta(s, v) \).

2. Values of \( d \) may only decrease. Therefore after \( d(v) \) reaches \( \delta(s, v) \), neither \( d(v) \) nor \( \pi(v) \) change.

Shortest path tree proof

Consider any (properly initialized) relaxation based algorithm \( A \) for the single-source shortest paths finding (in particular, Bellman-Ford). Let us prove the following simple property:

**Lemma 0.2** If a relaxation on edge \( (u, v) \) decreases \( d(v) \) to \( d(u) + c(u, v) = \delta(s, v) \), at that moment \( d(u) \) already equals \( \delta(s, u) \).
Proof: The inequality to one direction is given by Fact 0.1(1). Assume now to the contrary that the cost of an optimal path $P$ from $s$ to $u$ is strictly lesser than the current value of $d(u)$. Then, the path $P \cdot (u,v)$ costs less than $d(u) + c(u,v) = \delta(s, v)$, a contradiction. \qed

At any moment of an execution of $A$, denote by $\bar{V}$ the vertex set $\{v \in V : d(v) = \delta(s, v) < \infty\}$. By Fact 0.1(2), $\bar{V}$ may only grow during the execution.

Proposition 0.3 At any moment, the graph $T$ formed by the vertices in $\bar{V}$ and the back-pointers from them is a shortest path tree from $s$ to the vertices in $\bar{V}$.

Proof: We prove by induction on adding vertices to $\bar{V}$. The basis case $\bar{V} = \{s\}$ is trivial.

Assume that $T$ is a shortest path tree to $\bar{V}$, when the relaxation on edge $(u, v)$ decreases $d(v)$ to $d(u) + c(u, v) = \delta(s, v)$. By Lemma 0.2, $u$ is in $T$. By the induction assumption, the path from $s$ to $u$ in $T$, $P_u$, costs $\delta(s, u)$. As a result of the relaxation, the new vertex $v$ and the leaf edge from $u$ to $v$ are added to $T$, keeping it be a tree rooted at $s$. The path $P_u \cdot (u,v)$ to $v$ in $T$ costs $\delta(s, u) + c(u,v) = d(u) + c(u,v) = \delta(s,v)$. Hence, the new $T$ is a shortest path tree from $s$ to the new $\bar{V}$. By Fact 0.1(2), back-pointers from vertices currently in $\bar{V}$ will never change after that. \qed

This Proposition implies straightforwardly that $A$ builds the shortest paths tree from $s$ to all vertices that it eventually provides the optimal value of $d$ to.

Note that the correctness of computing distances from $s$ is usually proved independently of the shortest paths tree property. In particular, based on the classic proof that BF computes the distances correctly for a graph with no negative cycle, Proposition 0.3 provides that it computes correctly also the shortest paths tree for such graphs.

References

