# Shortest Paths in Directed Graphs

Jonathan Turner

January 22, 2013

This note is adapted from *Data Structures and Network Algorithms* by Tarjan.

Let $G = (V, E)$ be a directed graphs and let *length* be a real-valued function on $E$. The *length of a directed path* in $G$ is defined to be the sum of the lengths of its edges. So for example, in Figure 1, the length of the path $c$, $a$, $b$, $g$ is 14. A *shortest path* joining a pair of vertices $u$ and $v$ is

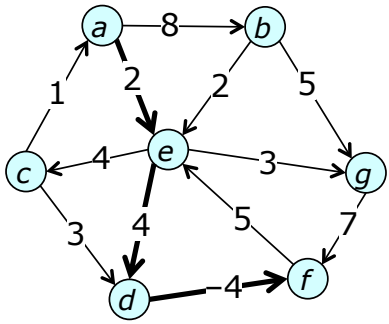


Figure 1: Example of a shortest path in a graph

defined as a path of minimum length among all paths from $u$ to $v$. So for example, the highlightes path $a$, $e$, $d$, $f$ in Figure 1 is a shortest path from $a$ to $f$, but the path $a$, $b$, $g$, $f$ is not.

The shortest path problem has many applications. For example, it is used to determine the best way to forward packets in the internet, and by mapping applications to determine the shortest driving distance between two locations. It also arises in less obvious forms, and often appears as a subproblem within other problems.

Note that we allow edges to have negative lengths, which means that shortest paths may have negative lengths as well. This allows the possibility that a graph might contain a cycle of negative length, in which case shortest

paths are not well-defined (since we can get a path of arbitrarily small length by traversing a negative cycle multiple times). This means that we have to take extra care when dealing with graphs that have negative edge lengths.

You might well wonder why we would want to allow negative edge lengths. While many applications of shortest paths do not require negative edge lengths, there are other applications where they are useful or even essential. Indeed, shortest path computations are used in algorithms for the *min cost flow* problem and in this context, negative edge lengths are an unavoidable feature of the problem.

We note that if there is no negative cycle on any path from $s$ to $t$ in a graph, then there must be a *simple path* of minimum length, since we can always remove a simple cycle of non-negative length from a path without increasing its length. So for example, in Figure 2 the shortest path $s$, $a$, $b$, $c$, $t$ can be replaced with the simple shortest path $s$, $a$, $t$ by removing the non-negative cycle $a$, $b$, $c$. These observations yield the following basic

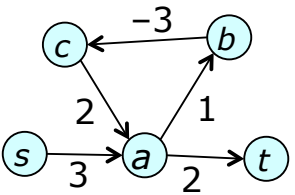


Figure 2: Removing non-negative cycles from shortest paths

theorem.

**Theorem 1** *Let $G$ be digraph with a path from $s$ to $t$. There is a shortest path from $s$ to $t$ if and only if no path from $s$ to $t$ that contains a negative cycle. If there is a shortest path from $s$ to $t$, there is one that is simple.*

There are several variants of the shortest path problem. The simplest is the *single pair problem*, in which we are given a source vertex $s$ and a sink vertex $t$ and our objective is find a shortest path from $s$ to $t$. In the *single source problem*, we are given a source vertex $s$, and our objective is to find shortest paths from $s$ to every other vertex in $G$. In the *single sink problem*, we are given a sink vertex $t$, and our objective is to find shortest paths to $t$ from every other vertex in $G$. Finally, in the *all pairs problem*, our objective is to find shortest paths between every pair of vertices. It turns out that solving the single pair problem generally requires solving a single source (or single sink) problem. Also, the single source and single sink problems are

essentially the same, and the all pairs problem can be solved by solving the single source problem multiple times. Hence, it makes sense to focus most of our attention on the single source version of the problem.

Solutions to the single source problem can be expressed conviently using a *shortest path tree*. A *directed spanning tree* of a graph is a directed subtree of the graph that includes a directed path from the root to every vertex. A shortest path tree is a directed spanning tree rooted at the source vertex $s$, in which all paths are shortest paths. So for example, the heavy weight edges in Figure 3 form a shortest path tree rooted at $a$. The following theorem

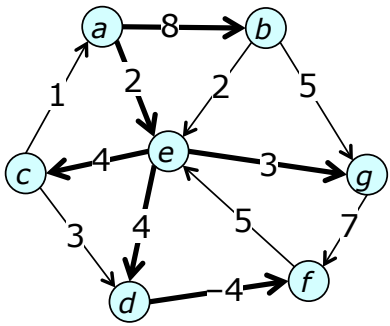


Figure 3: Shortest path tree

characterizes the existence of shortest paths in terms of shortest path trees.

**Theorem 2** *$G$ contains shortest paths from $s$ to every other vertex if and only if it contains a shortest path tree with root $s$.*

The proof of this will be deferred until later.

Next, we describe a key property of shortest path trees. If $T$ is a spanning tree of $G$ with root $s$, define $\mathit{distance}(v)$ to be the length of the path from $s$ to $v$ in $T$.

Consider an edge $(v, w)$ in $G$ that is not in $T$ and note that that if $\mathit{distance}(w) > \mathit{distance}(v) + \mathit{length}(v, w)$, then the tree path from $s$ to $w$ is not a shortest path (see Figure 4). Hence, if $T$ is a shortest path tree, then we must have $\mathit{distance}(w) \leq \mathit{distance}(v) + \mathit{length}(v, w)$ for all edges $(v, w)$ in $G$.

What about the converse? That is, can we say that if $\mathit{distance}(w) \leq \mathit{distance}(v) + \mathit{length}(v, w)$ for all edges $(v, w)$ in $G$, that all paths in $T$ are shortest paths?

To prove this, we must show that for any path $p$ in $G$ from $s$ to a vertex $x$, $\mathit{distance}(x) \leq \mathit{length}(p)$. Note that if $p$ consists of a single edge $(s, x)$,

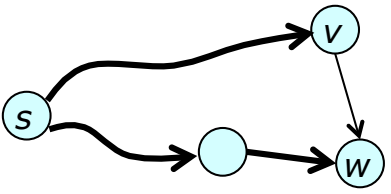


Figure 4: Shortest path tree property

then

$$length(p) = distance(s) + length(s, x) \geq distance(x)$$

So, let's try to prove the general result using induction on the number of edges in $p$. Assume then that $p$ has $k + 1$ edges and that for all vertices $x'$ and all paths $p'$ with $k$ edges, $distance(x') \leq length(p')$. Now, let $q$ be the path consisting of the first $k$ edges of $p$ and let $y$ be the last vertex in $q$ (see Figure 5). Then,

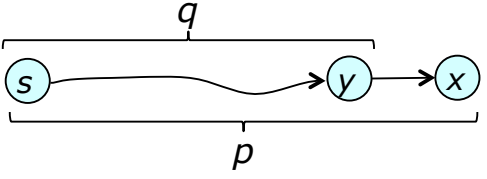


Figure 5: Inductive step establishing shortest path tree property

$$\begin{aligned} distance(x) &\leq distance(y) + length(y, x) \\ &\leq length(q) + length(y, x) \\ &= length(p) \end{aligned}$$

This gives us the following theorem.

**Theorem 3** *Let $T$ is a spanning tree of $G$ with root $s$, and let $distance(v)$ be the length of the path from $s$ to $v$ in $T$. $T$ is a shortest path tree if and only if, $distance(w) \leq distance(v) + length(v, w)$ for all edges $(v, w)$ in $G$.*

The property described in the theorem is called the *shortest path tree property* and is the basis for a fundamental method for solving the shortest path problem. This method essentially converts an arbitrary spanning tree into a shortest path tree by finding edges that violate the shortest path tree property and modifying the tree so as to eliminate the violations. This

method underlies the various specific algorithms that are used to find shortest paths.