# Johnson's Algorithm

## Weighted Directed Graphs

Let $G = (V, E)$ be a directed graph. A *weight function* of $G$ is a function $w : E \to \mathbb{R}$. We say the ordered pair $(G, w)$ is a *weighted graph.* The *shortest path* problem is to find the path from $x$ to $y$ of smallest total weight, for $x, y \in V$, The *single pair* shortest path problem is to find the minimum weight path for a single pair $(x, y)$. The *single source* shortest path problem is to find minimum weight paths from a specified source vertex to all vertices, while the *all pairs* shortest path problem is to find minimum weight paths for every choice of $(x, y)$.

## Equivalent Weightings

Two weight functions, $w_1$ and $w_2$ on a directed graph $G = (V, E)$ are *equivalent* if there is a function $h : E \to \mathbb{R}$ such that $w_2(x, y) = w_1(x, y) + h(x) - h(y)$ for all $(x, y) \in E$.

**Theorem 1** *If $w_1$ and $w_2$ are equivalent weight functions on a directed graph $G = (V, E)$, and $x, y \in V$, any shortest path from $x$ to $y$ in $(G, w_1)$ is also a shortest path from $x$ to $y$ in $(G, w_2)$.*

## Johnson's Algorithm

Johnson's algorithm solves the all-pairs shortest path problem for a weighted directed graph $(G, w)$ with no negative weight cycles. Write $G = (V, E)$, let $n = |V|$ and $m = |E|$. The time complexity of Johnson's algorithm is $O(nm \log n)$, which is less than the $\Theta(n^3)$ time complexity of the Floyd-Warshall algorithm, provided $m$ is small enough.

The first step of Johnson's algorithm is to create the augmented weighted directed graph, $(G^*, w^*)$. $G^*$ has one new vertex, $s$, and $n$ new arcs, $\{(s, x) : x \in V\}$, where $w^*(x, y) = (x, y)$ if $(x, y) \in E$, and $w^*(s, x) = 0$. We then use the Bellman-Ford algorithm to run the single source shortest path problem on $(G^*, w^*)$ For all $x \in V$, let $h(x)$ be the least weight of any path in $G^*$ from $s$ to $x$. Since there is an arc of weight zero from $s$ to $x$, we have $f(x) \leq 0$. We now define $w'(x, y) = w(x, y) + h(x) - h(y)$, and solve the all-pairs shortest path problem on $(G, w')$

**Theorem 2** *$w'(x, y) \geq 0$ for all $(x, y) \in E$.*

*Proof:* Since $f$ is the solution to the single source shortest path problem on $G^*$, we have $f(y) \leq f(x) + w(x, y)$. Thus $w'(x, y) = w(x, y) + f(x) - f(y) \geq 0$, $\blacksquare$

Since $w'$ is never negative, we can use Dijkstra's algorithm $n$ times to solve the single source shortest path problem on $(G, w')$ using each vertex as the source, giving us the function $dist'(x, y)$

for any $x, y \in V$. We then define $dist(x, y) = dist'(x, y) - f(x) + f(y)$ to obtain the solution to the original problem.

## A Small Example

Let $(G, w)$ be the weighted directed graph shown in Figure 1, where $n = 7$ and $m = 9$. There are no negative cycles, but there are negative arcs.

Since $m$ is considerably less than $\dfrac{n^2}{\log n}$ we expect Johnson's algorithm to be faster than the Floyd-Warshall algorithm.

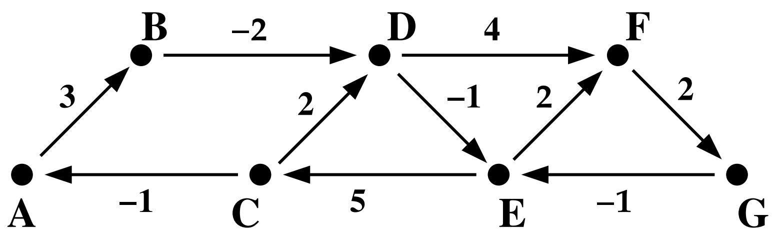


**Figure 1:** $(G, w)$, a Weighted Directed Graph.

We augment $G_1$ by creating a new vertex $s$ and an arc of length zero from $s$ to each vertex of $G$; these new arcs are shown in red in Figure 2. We call the resulting directed graph $G^*$. We apply the Bellman-Ford single source algorithm to $G^*$. For each vertex $x$ of $G$, let $f(x)$ be the minimum weight of any path in $G^*$ from $S$ to $x$. The values of $f$ are shown in red in Figure 2.

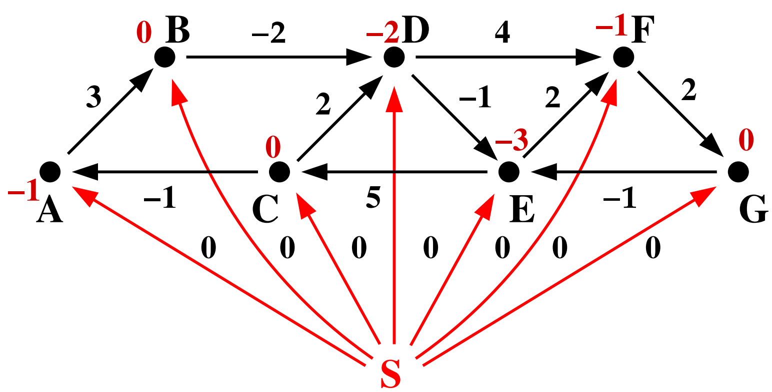


**Figure 2:** The Augmented Weighted Directed Graph $G^*$.

We now compute the adjusted weights, $w'(x,y)$ for any vertices $x$ and $y$. The definition of $w'$ is:

$$w'(x,y) = w(x,y) + f(x) - f(y)$$

Let $(G, w')$ is a weighted directed graph with no negative weight arcs. We show the adjusted weights in Green in Figure 3.

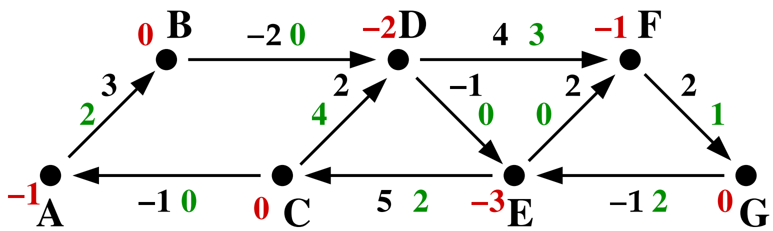


**Figure 3:** Calculation of Adjusted Weights $w'$ on $G$

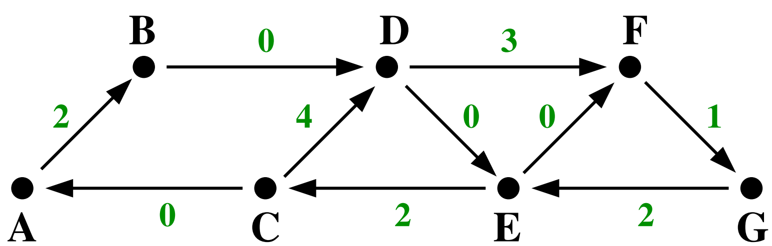


**Figure 4:** The Weighted Directed Graph $(G, w')$

We now run Dijkstra's algorithm on $(G, w')$ $n$ times. For each run we pick one vertex of $G$ to be the source. Each run yields a tree of shortest paths rooted at the chosen vertex, which we call the Dijkstra tree.

In Figure 5 we show the $n$ Dikstra trees. Minimum path weight values are written in dark red.

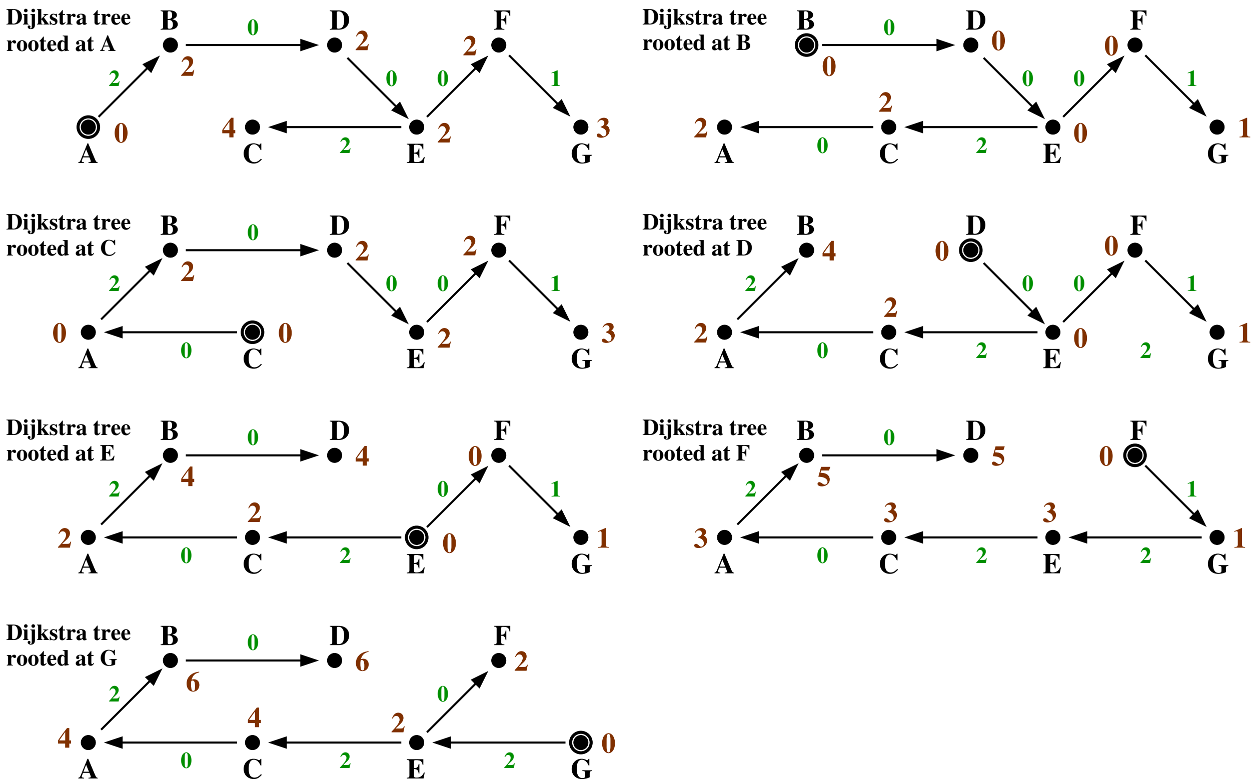


**Figure 5:** Dijkstra Trees for each Choice of Source Vertex.

In Figure 6 we replace the adjusted weight by the original weight for each arc. We relabel the arcs of each Dijkstra tree. The true minimum path from $x$ to $y$ is unique path from $x$ to $y$ in the tree rooted at $x$. Weights of those minimum paths are shown in red.

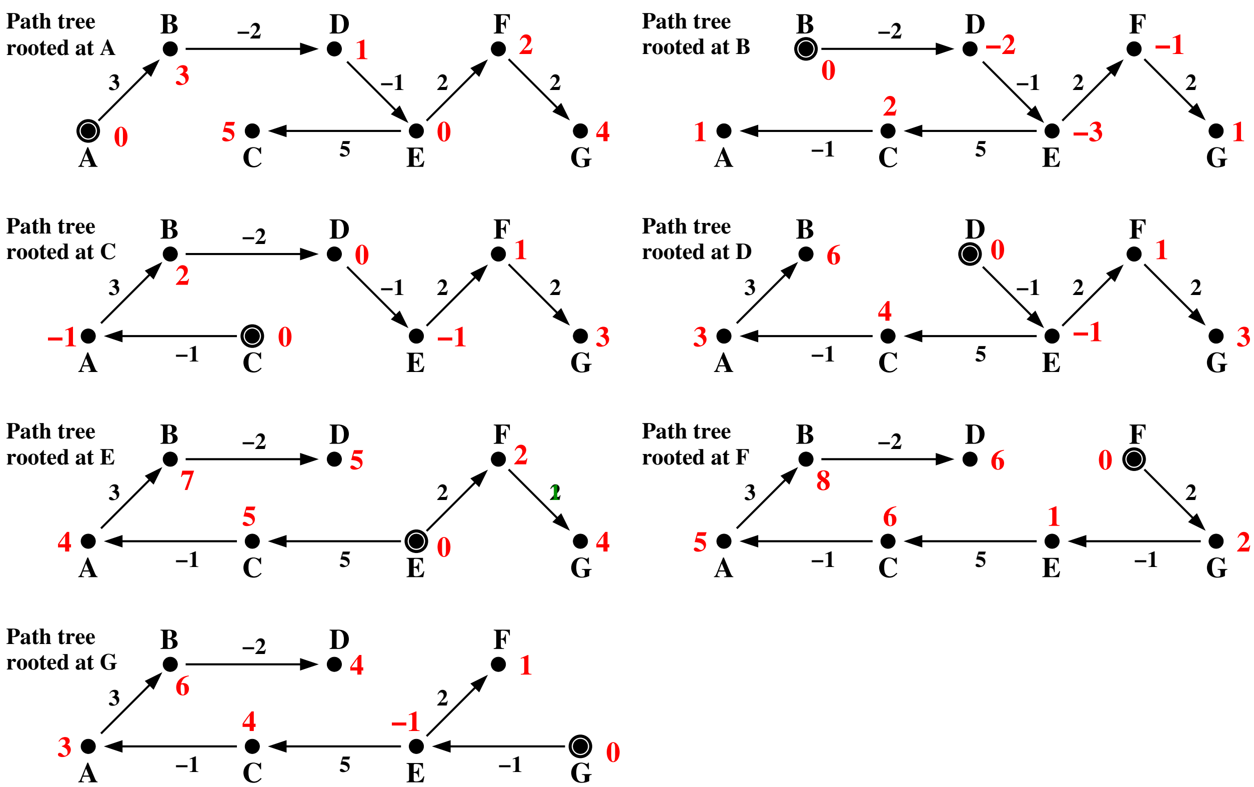


**Figure 6:** Shortest Path Weights for $(G, w)$

| | A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|---|
| $A$ | 0 | 3<br>A | 5<br>E | 1<br>B | 0<br>D | 2<br>E | 4<br>F |
| $B$ | 1<br>$C$ | 0 | 2<br>$E$ | $-2$<br>$B$ | $-3$<br>$D$ | $-1$<br>$E$ | 1<br>$F$ |
| $C$ | $-1$<br>$C$ | 2<br>$A$ | 0 | 0<br>$E$ | $-1$<br>$D$ | 1<br>$E$ | 3<br>$F$ |
| $D$ | | | | | | | |
| $E$ | | | | | | | |
| $F$ | | | | | | | |
| $G$ | | | | | | | |

We now write the array showing the results. The minimum weight of a path from $x$ to $y$ is in row $x$ and column $y$. Underneath that weight is the back pointer.

**Exercise:** Fill in the missing information in the array.