Min-"weight" Path Algorithm (i.e., Dijkstra's algorithm)

Let G = (V,E) be a digraph (or graph) with n vertices. For specificity we assume that $V = \{v_1, v_2, \dots, v_n\}$, and suppose that there is a *weight* w_{ij} (a positive real number) associated with each edge (v_i,v_j) — this might be a geographic distance, a transportation cost, etc. We seek to determine a *minimum weight* path from one node to another, or determine if no path exists. This is an alternative way to think about Dijkstra's algorithm, but it is effectively the same process described in our text. This algorithm uses a BFS strategy and embellishes the computation of the sequence of subsets developed previously in the min-step algorithm for $P_k \subseteq V$:

$P_0 = \{u\}$, and

$P_{k+1} = P_k \cup \{y \mid x \in P_k \text{ and } (x,y) \in E\}, k \geq 0.$

The idea is to additionally record the minimum path weight for each of the subsequent length paths as we go. However, a longer path may be "lighter" than a shorter one if its edge weights are sufficiently smaller — hence we cannot terminate the process the first time the target node is encountered. But there are finitely many alternatives to consider as the process progresses stage by stage, and we can stop by stage n–1 since a minimum weight path will never contain a cycle. Suppose that we seek a minimum weight path from v_p to v_q.

Let

$W_0 = \{(v_p,0)\}$, and

$W_{k+1} = \{(v_j,w) \mid w = \min(\{w_j\} \cup \{w_i+w_{ij} \mid (v_i,w_i) \in W_k \text{ and } (v_i,v_j) \in E\}),$
where $(v_j,w_j) \in W_k\}, k \geq 0.$

Formally, we have the

Assertion: W_k is the set of all pairs (v_j,w_j) where v_j is reachable from v_p by a path of length k or less and w_j is the minimum weight of all such paths.

We observe that since the weights are non-negative, there is always a min-weight path that is elementary — deleting a cycle from a path will never increase the weight. Therefore a min-weight path of length at most n–1 can be found if there is any path at all. Hence we inspect W_{n-1} (as in the min-length algorithm if $W_{k+1} = W_k$, then $W_k = W_{n-1}$) for a pair (v_q,w) and w is the min-weight of paths from v_p to v_q; if no such pair belongs to W_{n-1}, then v_q is unreachable from v_p.

Again this process determines the min-*weight* of a path, not the path itself. But again the path can be extracted by a traceback process. If a min-weight path from v_p to v_q has weight w, and we stop at the node immediately before v_q, say v_j, then we must have a min-weight path from v_p to v_j, say of weight w'. But then $w = w'+w_{jq}$, so from (v_q,w) we can identify the node v_j (i.e., $w' = w - w_{jq}$) and then repeat this process to trace the nodes back to v_p.