1. Correctness of Dijkstra's algorithm

<u>Thm.</u> (4.2 in Baase, 1988).

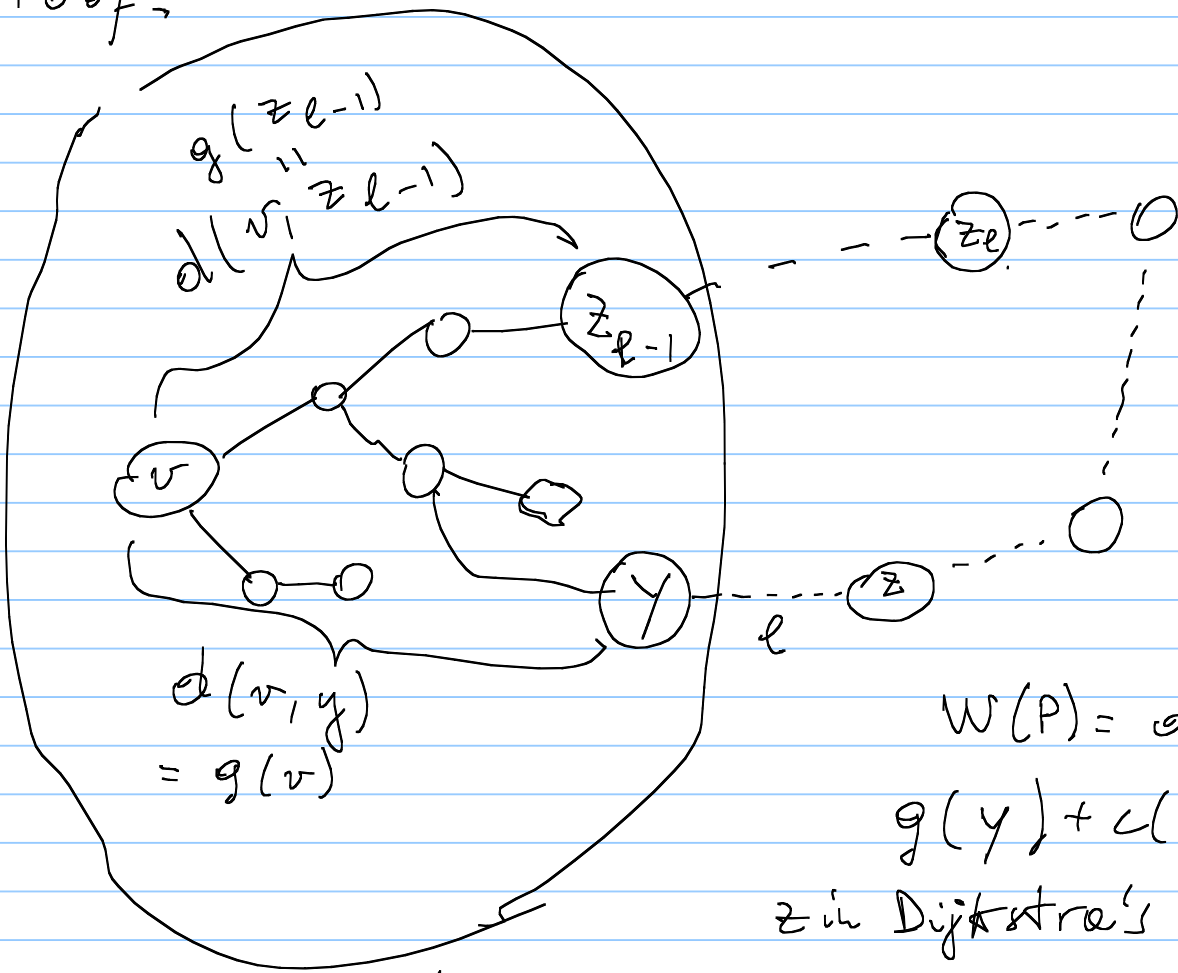
$G = (V, E, W)$, W are weights in $\mathbb{Z}^+$.

$V' \subset V$, $v \in V'$.

If e is chosen to minimize $d(v, \text{tail}(e)) + W(e)$ over all edges with one vertex in V' and one in $V \setminus V'$, then the path consisting of e adjoined to the end of a shortest path from e to $\text{tail}(e)$ is a shortest path from v to $\text{head}(e)$.

In other words: When Dijkstra's algorithm closes a node, it has found a shortest path to that node.

Proof.



Closed nodes $= V'$

$e = yz$

Dijkstra's algorithm chooses e, and $P = v \dots yz$.

Let $P' = v \dots z_{\ell-1} z_\ell \dots z$ be any other path from v to z.

$W(P) = d(v, y) + W(e) =$

$g(y) + c(y, z) \leq$ (by the choice of z in Dijkstra's algorithm) $\leq$

$g(z_{\ell-1}) + c(z_{\ell-1}, z_\ell) = d(v, z_{\ell-1}) + W(z_{\ell-1} z_\ell) =$

$= W(P')$.

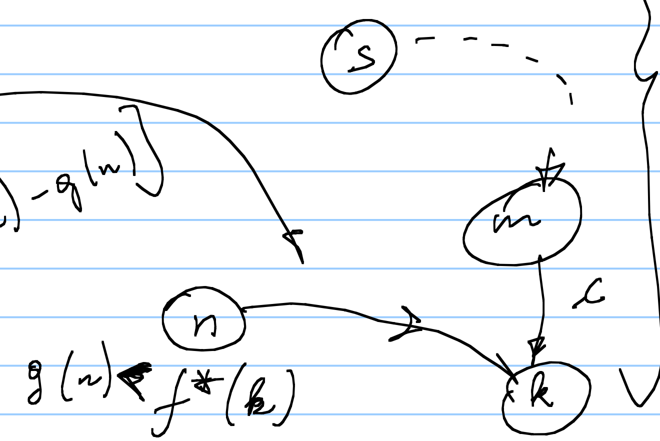
2. Weak optimality of Dijkstra's algorithm

Dijkstra's algorithm expands the least number of nodes among all blind, unidirectional algorithms that solve the single-source, single target shortest path problem and do comparisons of weights.

Proof. By an oracle (adversary) argument

$f^*(p)$ = length of shortest path to node p

edge introduced by the adversary, of cost $c(n,k) = \frac{1}{2}[f^*(k) - g(n)]$



solution found by impostor algorithm B, which does not close node n

3. Strong optimality of Dijkstra's algorithm when implemented with Fibonacci heaps. $\Theta(n \log n + m)$

Dijkstra's algorithm produces a shortest-path spanning tree of G <u>and</u> a sequence of all vertices in G ordered by non-decreasing distance from s.

Sorting can be transformed trivially to the problem solved by Dijkstra's algorithm.