Shortest Path Problem

Shortest path network.
- Directed graph $G = (V, E)$.
- Source $s$, destination $t$.
- Length $l_e$ = length of edge $e$.
- $l_e > 0$ for all edges $e$.

Shortest path problem: (the Google Maps problem!) find shortest path from $s$ to $t$.

Dijkstra's Algorithm: Implementation

Dijkstra's Algorithm $(G, s)$ {
    $S = \{s\}$ // $S$ is the set of explored nodes
    $d(s) = 0$ // $d$ is the distance to the node from $s$
    while $S \neq V$ {
        // there are unexplored nodes
        select a node $v$ from $V-S$ with an edge from $S$ for
        which the distance from $s$ to $v$ is the minimum of all
        paths to any node in $V-S$
        add $v$ to $S$
        $d(v) = \text{minimum distance from } s \text{ to } v$
    }
}

How do we implement this efficiently?
Dijkstra's Algorithm \((G, s)\) {
\[S = \{s\}\]  // \(S\) is the set of explored nodes
\[d(s) = 0\]  // \(d\) is the distance to the node from \(s\)
lastNode = s

while \(S \neq V\) {  // there are unexplored nodes
    for each edge \((\text{lastNode}, v)\) where \(v\) is in \(V - S\) {
        \[\text{dist}_v = d(\text{lastNode}) + l(\text{lastNode}, v)\]
        if \(d'(v)\) is unknown {
            \[d'(v) = \text{dist}_v\]
            heap.addElement \((v, d'(v))\)
        } else if \(\text{dist}_v < d'(v)\) {
            \[d'(v) = \text{dist}_v\]
            heap.changeKey \((v, d'(v))\)
        }
    }
    lastNode = heap.extractMin()
    add lastNode to \(S\)
    \[d(\text{lastNode}) = d'(\text{lastNode})\]
}

Cost?

Dijkstra's Algorithm \((G, s)\) {
\[S = \{s\}\]  // \(S\) is the set of explored nodes
\[d(s) = 0\]  // \(d\) is the distance to the node from \(s\)
lastNode = s

while \(S \neq V\) {  // there are unexplored nodes
    for each edge \((\text{lastNode}, v)\) where \(v\) is in \(V - S\) {
        \[\text{dist}_v = d(\text{lastNode}) + l(\text{lastNode}, v)\]
        if \(d'(v)\) is unknown {
            \[d'(v) = \text{dist}_v\]
            heap.addElement \((v, d'(v))\)
        } else if \(\text{dist}_v < d'(v)\) {
            \[d'(v) = \text{dist}_v\]
            heap.changeKey \((v, d'(v))\)
        }
    }
    lastNode = heap.extractMin()
    add lastNode to \(S\)
    \[d(\text{lastNode}) = d'(\text{lastNode})\]
}

How do we prove this correct?

Edsger W. Dijkstra

The question of whether computers can think is like the question of whether submarines can swim.

A. Nico Habermann

- First dean of the School of Computer Science at CMU
- CRA A. Nico Habermann Award given to a person making outstanding contributions to increasing participation of underrepresented minorities in CS