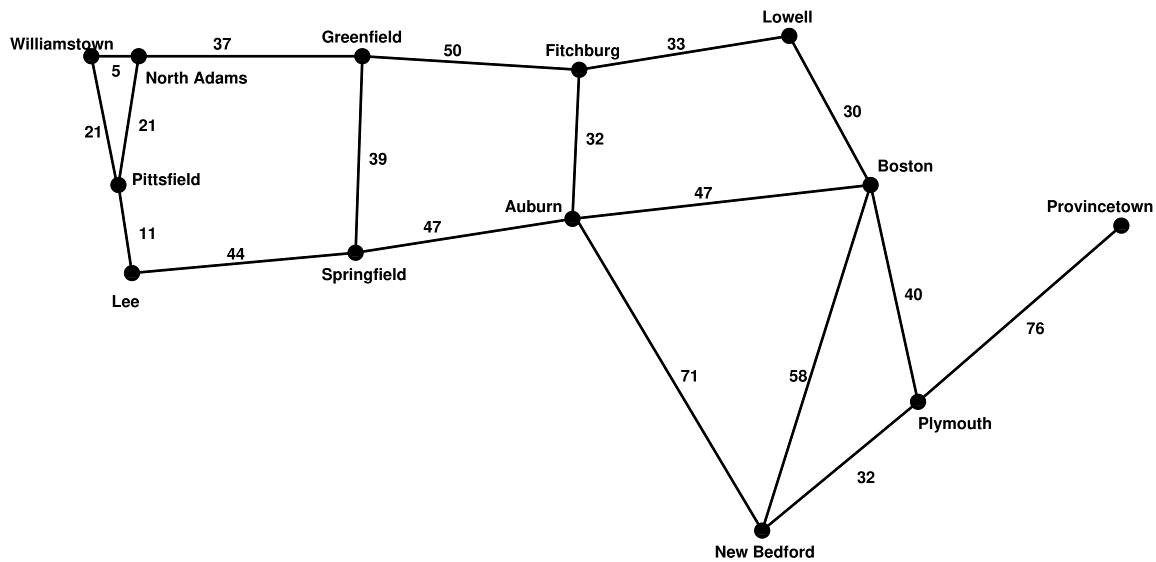
Computer Science 385
Design and Analysis of Algorithms
Siena College
Spring 2018

Topic Notes: Dijkstra's Algorithm Example

Pseudocode:

ALGORITHM DIJKSTRA(G, s)
  //Input: $G = (V, E)$ a graph with weighted edges
  //Input: $s \in V$, the starting vertex
  $T \leftarrow$ an empty map
  $PQ \leftarrow$ an empty priority queue
  // mark all vertices in V as unvisited
  for all $v \in V$ **do**
    $v.visited \leftarrow$ **false**
  // s is found at distance 0
  $T.add(s, (0, null))$
  $s.visited \leftarrow$ **true**
  // add each edge from s to PQ
  for all $(s, v) \in E$ **do**
    $PQ.add((s, v), (s, v).cost)$
  // main loop
  while $T.size < G.size$ **and not** $PQ.empty$ **do**
    // remove edges from PQ until empty or we find one connecting
    // a visited vertex to an unvisited vertex
    repeat
      $(u, v) \leftarrow PQ.remove$
    until $u.visited \neq v.visited$ **or** $PQ.empty$
    // WLOG, assume u is visited (*i.e.*, is in T) and v is
    // unvisited (not in T), meaning we found a way to v
    $vcost \leftarrow T.get(v).cost + (u, v).cost$
    $T.add(v, (vcost, (u, v)))$
    $v.visited \leftarrow$ **true**
    // for each w, a vertex adjacent to v
    for all $(v, w) \in E$ **do**
      if $w.visited =$ **false then**
        $PQ.add((v, w), vcost + (v, w).cost)$
  return T

Graph to work with:



Result "tree"/table/map:

Place	(distance,last-edge)
W'town	(0, `null`)

Priority queue:

(distance,last-edge)	Seq