Design and Analysis of Algorithms — Lab

R S Milton
Department of CSE, SSN College of Engineering

Session 7: Greedy Algorithms
Dijkstra's Algorithm
13 February 2020

1 MinHeap

The goal is to implement an Abstract Data Type Minheap with the operations INSERT, DELETEMIN and DECREASEKEY.

Develop the program in the following steps.

1. Define a class Heap . It should have an array field A to store the keys (integers) of the heap and a field n to maintain the heap size, the actual number of keys in the heap. We decide to store the keys of the heap starting from index 1. Therefore, n is both the size of the heap as well as the position of the last key in the heap. Define a suitable constructor. Define a method print(i) that prints the heap rooted at position i displaying the tree structure. Define a class HeapDemo . Initialize the heap with a list of integers satisfying min-heap property, and print them using print() .

2. The Heap class should provide the following methods:

   (a) Insert(i, k): Inserts key item `i` with `k` into the heap.

   (b) DeleteMin() : Deletes the minimum key from the heap and returns it.

   (c) DecreaseKey(x, k) : Updates the key of item `x` to `k`.

2 Single Source Shortest Paths

1. Design and implement `DirectedEdge` class.

```
DirectedEdge(v, w, weight)
weight()
from()
to()
```

2. Design and implement `WeightedDigraph` class.

```
WeightedDigraph(n)          # empty n-vertex digraph
EdgeWeightedDigraph(data) # construct from data
V()                         # number of vertices
E()                         # number of edges
add_edge(e)                 # add e to this digraph
adj(v)                      # out edges from v
edges()                     # all edges in this digraph
```

3. Design and implement `SP` class:

```
SP(G, s)      # constructor
distTo(v)     # distance from s to v, ∞ if no path
hasPathTo(v) # path from s to v?
pathTo(v)     # path from s to v, otherwise None
```

4. Add a method `relax(e)` to relax an edge `e` in `SP`.

5. Add a method `relax(G, u)` to relax all the tense out-edges of vertex `v` in the `WeightedDigraph` `G.`

6. Implement `DijkstraSP` algorithm as a method for the `WeightedDigraph`. Modify `relax(G, u)` and use it in `DijkstraSP`.

7. Test all your methods in each stage.