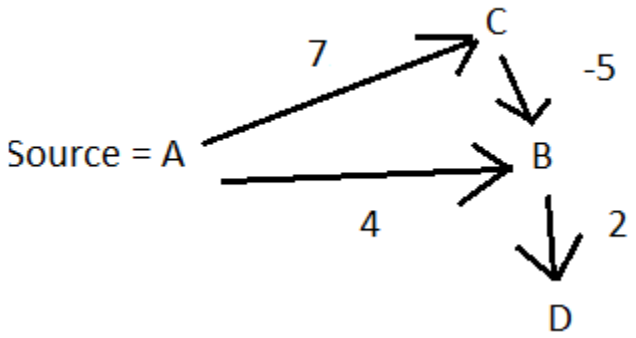
HW8 Theory

Alexander Kazantsev

March 2, 2016

Problem 6.6



First A is popped out of the queue and and edges (A, C) and (A, B) are relaxed. B is popped off and relax the edge (B, D) giving a distance of 6. D is now at the top, but has no edges. C is the last vertex left. The edge (C, B) is relaxed giving a total distance to B as 2. Now the queue is empty, but the shortest path to B from A is not correct.

Problem 7.1

```
Insert( val, v, adjList)
    assert v ∈ adjList
    if val !∈ adjList
        add(adjList, val)
        ListInsert(adjList[ v ], val)
    else
        ListInsert(adjList[ v ], val)
```

Delete(val, vertex,adjList)

assert val and vertex ∈ adjList

∀ e ∈ adjList [val]

if e == vertex

remove(adjList[vertex], e)

break

∀ e ∈ adjList [vertex]

if e == val

remove(adjList[val], e)

break

Problem 7.2

Restructuring the adjacency list as a matrix would allow for removal of the first edge in constant time. The down side is that this derivative requires more space than a linked list implementation.

	1	2	3	4	5	6	7
1	0	1	1	1	0	0	0
2	1	0	1	1	0	0	0
3	1	1	0	0	1	1	1
4	1	1	0	0	1	0	0
5	0	0	1	1	0	1	0
6	0	0	1	0	1	0	1
7	0	0	1	0	0	1	0

DeleteFirstEdge(vertex, adjMatrix)

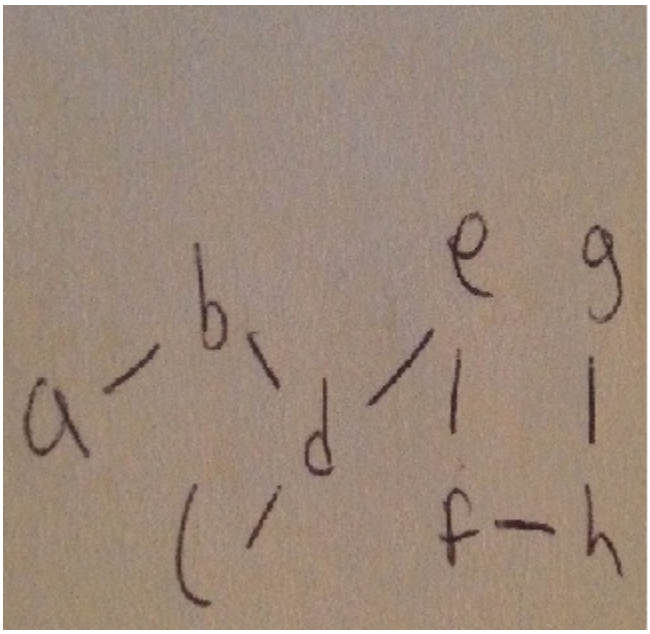
assert vertex ∈ adjList

adjMatrix[vertex[0]] [vertex] = Null

adjMatrix[vertex] [0] = Null

Problem 7.3

Part a



Part b