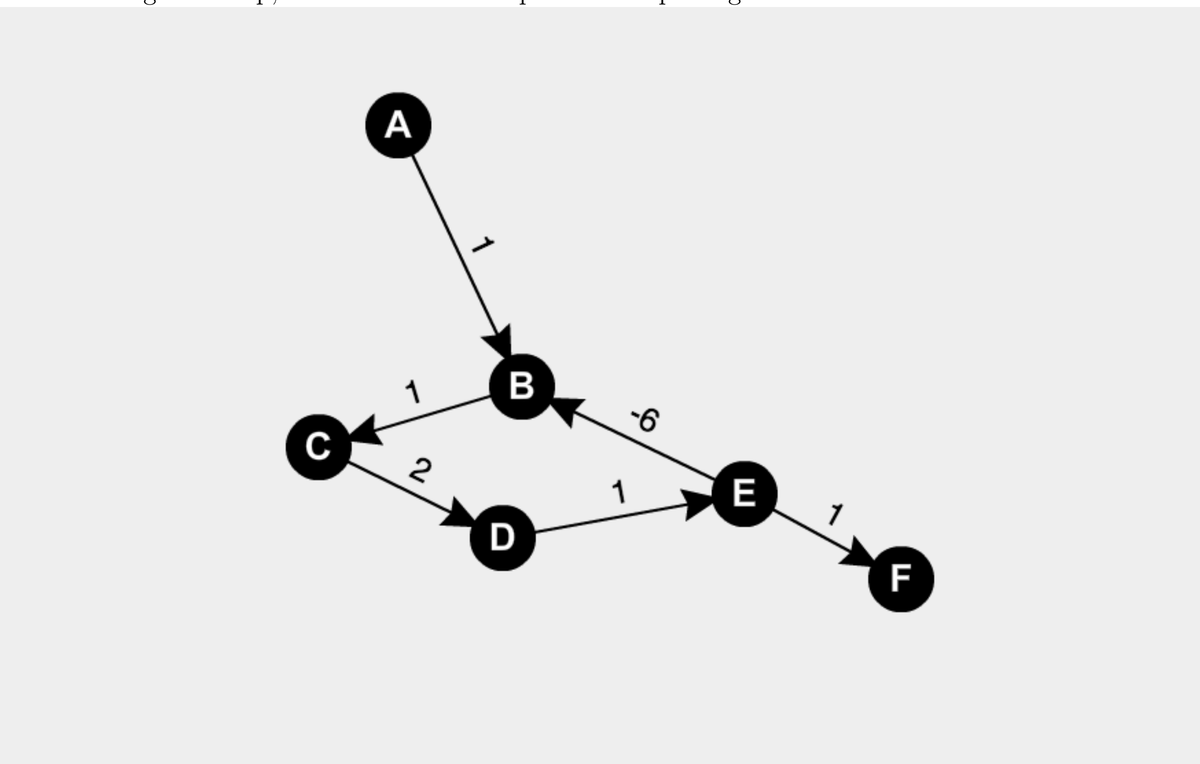
Algorithms and Data Structure 11

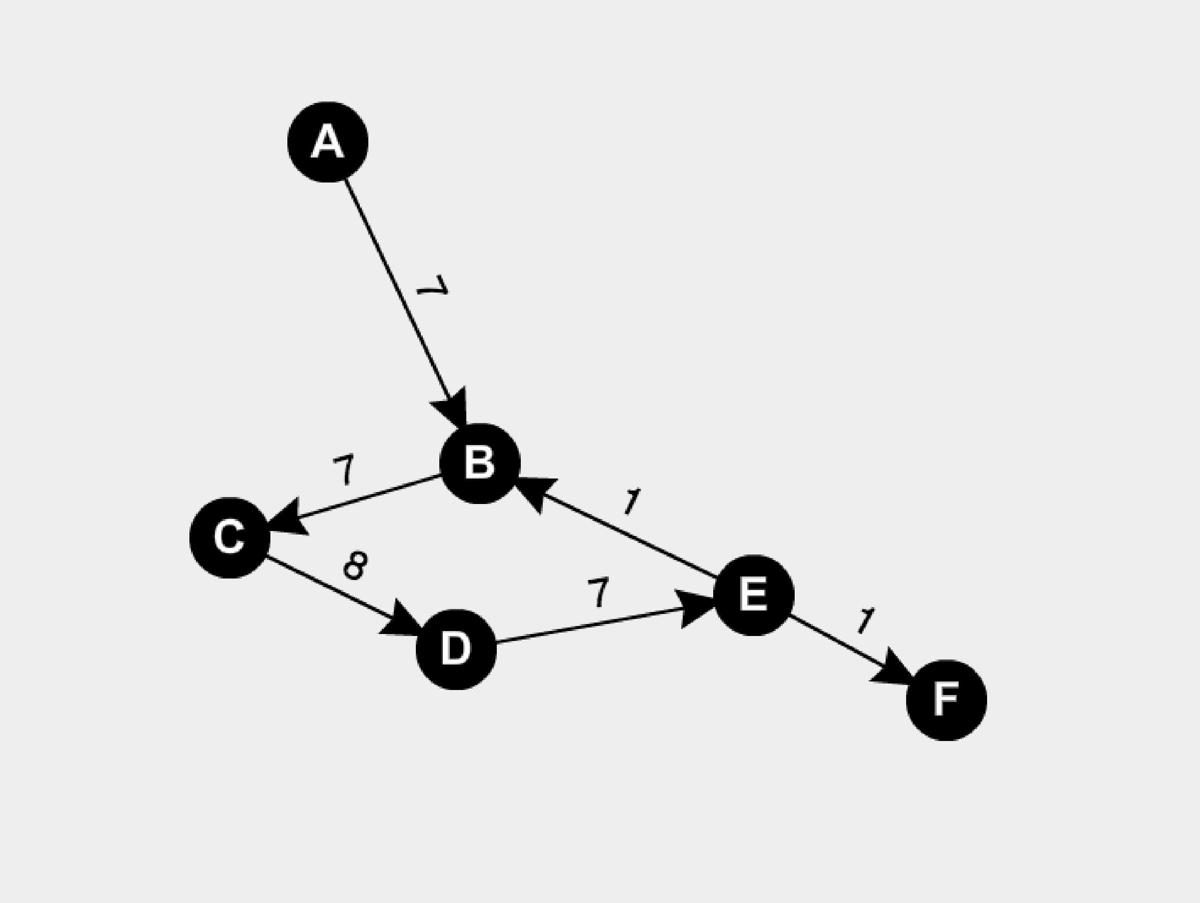
Yiping Deng

May 21, 2018

Problem 1

Such algorithm is obviously incorrect. Consider a graph with loop such that the sum of the loop is negative. Including the loop one more time will lead to a even lower path, thus we can conclude that if there is a negative loop, there is no shortest path. Example is given below:





Problem 2

```
from heapq import *

class Node:
    compare_by_y = False
    def __init__(self, index):
        self.dx = float('inf')
        self.dy = float('inf')
        self.index = index
        self.visited = False

    def __lt__(self, other):
        if Node.compare_by_y:
            return self.dy < other.dy
        return self.dx < other.dx
    def maxval(self):
        if(self.dx > self.dy):
            return self.dx
        return self.dy

def find_meetup_city(adj_matrix, your_city, friend_city):
    n = len(adj_matrix)
    vec = [Node(i) for i in range(n)]
    q = []

    # Dijkstra on x value
    vec[your_city].dx = 0
```

```
    heappush(q, vec[your_city])

    while q:
        v = heappop(q)
        for i in range(n):
            if i != v.index:
                altdist = v.dx + adj_matrix[v.index][i]
                if altdist < vec[i].dx:
                    vec[i].dx = altdist
                    heappush(q, vec[i])

    # Dijkstra on y value
    Node.compare_by_y = True
    vec[friend_city].dy = 0
    heappush(q, vec[friend_city])

    while q:
        v = heappop(q)
        for i in range(n):
            if i != v.index:
                altdist = v.dy + adj_matrix[v.index][i]
                if altdist < vec[i].dy:
                    vec[i].dy = altdist
                    heappush(q, vec[i])

    minCity = min(vec, key = Node.maxval)
    return minCity.index
```

Problem 3

a)

Let's first rephase our problem in math.

Let $G = (V, E)$ to be a directed graph such that every pair $u, v \in V, u \neq v$, either $(u, v) \in E$ or $(v, u) \in E$ is satisfied, but not all. Then there exists a finite sequence $(v_i)_{i \in \mathbb{N}}$ formed by set V such that $(v_i, v_{i+1}) \in E$.

Proof: The base case with $|V| = 2$ is trivial. There is only two possibility. $(v_1, v_2) \in E$ or $(v_2, v_1) \in E$. Each case we can generate a Hamiltonian path.

Suppose the statement holds for $|V| = n$, consider $|V'| = n+1$. Pick $v_0 \in V'$, define subgraph defined on $W = V' - v_0$. There is a ordering of W, denoted $w_1, ..., w_n$. Define $m = max(\{i : \forall j \leq i.(v_j, v_0) \in E\})$.

If such m is not well defined, we can place v_0 at the beginning, forming $v_0, w_1, w_2, ..., w_n$ as a Hamiltonian path.

If m is well defined, we can place v_0 as following:

$$w_1, w_2, ..., w_m, v_0, w_{m+1}, ..., w_n$$

b)

```
def picking_order(adj_matrix):
    rst = [] #storing the result

    for i in range(len(adj_matrix)):
        pos = 0
        for w in rst:
            if not adj_matrix[w][i]:
                break
            pos = pos + 1
```

```
        rst = rst[0:pos] + [i] + rst[pos:]
    return rst
```