Zijian Gao
26319560
4/7/2016
CS311 HW4

24.1-1.)

For lines 2-5:
Here I use the order given in Fig 24.4

1. (z,x) x.d=7 x.π=z
   (z,s) s.d=2 s.π=z
   (s,t) t.d=8 t.π=s
   (s,y) y.d=9 y.π=s
2. (t,x) nop
   (t,y) nop
   (t,z) nop
   (x,t) t.d=5 t.π=x
   (y,x) x.d=6 x.π=y
   (y,z) nop
   (z,x) nop
   (z,s) nop
   (s,t) nop
   (s,y) nop
3. (t,x) nop
   (t,y) nop
   (t,z) nop
   (x,t) t.d=4 t.π=x
   (y,x) nop
   (y,z) nop
   (z,x) nop
   (z,s) nop
   (s,t) nop
   (s,y) nop
4. (t,x) nop
   (t,y) nop
   (t,z) nop
   (x,t) nop
   (y,x) nop
   (y,z) nop
   (z,x) nop
   (z,s) nop
   (s,t) nop
   (s,y) nop

After the 4 iterations:
z.d=0 z.π=NIL
x.d=6 x.π=y
t.d=4 t.π=x
y.d=9 y.π=s
s.d=2 s.π=z

BELLMAN-FORD returns true starting at z.

Starting from z with w(z,x)=4

1. (z,x) x.d=4 x.π=z
   (z,s) s.d=2 s.π=z
   (s,t) t.d=8 t.π=s
   (s,y) y.d=9 y.π=s
2. (t,x) nop
   (t,y) nop
   (t,z) nop
   (x,t) t.d=2 t.π=x
   (y,x) nop
   (y,z) nop
   (z,x) nop
   (z,s) nop
   (s,t) nop
   (s,y) nop
3. (t,x) nop
   (t,y) nop
   (t,z) z.d=-2 z.π=t
   (x,t) nop
   (y,x) nop
   (y,z) nop
   (z,x) x.d=2 x.π=z
   (z,s) s.d=0 s.π=z
   (s,t) nop
   (s,y) y.d=7 y.π=s
4. (t,x) nop
   (t,y) nop
   (t,z) nop
   (x,t) t.d=0 t.π=x
   (y,x) nop
   (y,z) nop
   (z,x) nop
   (z,s) nop
   (s,t) nop
   (s,y) nop

Final values:
z.d=-2 z.π=t
x.d=2 x.π=z
t.d=0 t.π=x
y.d=7 y.π=s
s.d=0 s.π=z

The negative-weight cycle check on lines 5-7 will cause the algorithm to return false on the check for (t,z) because z.d > t.d+ w(t,z) or -2 > 0 - 4.

24.1-3.)

```
RELAX(u,v,w):
   if v.d>u.d + w(u,v)
      v.d = u.d+w(u,v)
      v.π =u
      return true
   else return false

BELLMAN-FORD(G,w,s):
   INITIALIZE-SINGLE-SOURCE(G,s)
   changed = false
   while (changed)
      changed = false
      for each edge (u,v) in G.E
         changed = RELAX(u,v,w)
   for each edge (u,v) in G.E
      if v.d > u.d + w(u,v)
         return false
   return true
```

Since it is unknown what m is, all edges must be relaxed to their final shortest-path values, which will take m iterations. On the m+1th iteration, no values will change by the upper-bound property and the algorithm will terminate. The original BELLMAN-FORD makes V-1 iterations, which may make more iterations than *m+1* if *m*+1 < V-1.

24.3-1.)
Assuming same-priority nodes are disambiguated in ascending alphabetical order.

Vertex s as source:

1. u = s
   S = {s}
   (s,t) t.d=3 t.π=s
   (s,y) y.d=5 y.π=s
2. u = t
   S = {s,t}
   (t,x) x.d=9 x.π=t
   (t,y) NOP
3. u = y
   S = {s,t,y}
   (y,t) NOP
   (y,x) NOP
   (y,z) z.d=11 z.π=y
4. u = x
   S = {s,t,y,x}
   (x,z) NOP
5. u = z
   S = {s,t,y,x,z}

(z,s) NOP
(z,x) NOP

Shortest paths:

z.d=11 z.π=y
x.d=9 x.π=t
y.d=5 y.π=s
t.d=3 t.π=s
s.d=0 s.π=NIL

Vertex z as source:

1. u = z
   S = {z}
   (z,s) s.d=3 s.π=z
   (z,x) x.d=7 x.π=z
2. u = s
   S = {z,s}
   (s,t) t.d =6 t.π=s
   (s,y) y.d=8 y.π=s
3. u = t
   S = {z,s,t}
   (t,y) NOP
   (t,x) NOP
4. u = x
   S = {z,s,t,x}
   (x,z) NOP
5. u = y
   S = {z,s,t,x,y}
   (y,t) NOP
   (y,x) NOP
   (y,z) NOP

Shortest Paths:

y.d=8 y.π=s
t.d=6 t.π=s
x.d=7 x.π=z
s.d=3 s.π=z
z.d=0 z.π=NIL