Dijkstra’s Method

overview

- single source shortest path
- no negative edge weights

Start with node s at distance 0

- S=∅ will be the set of nodes whose distances are known
- all other nodes have distance ∞

repeatedly

- find node u ∈ V-S whose shortest path estimate is minimum
- add u to S
- relax all edges leaving u

relaxing an edge

```
relax(u,v)
     if u.dist +W[u,v] < v.dist
          then
               v.dist = u.dist + W[u,v]
               v.prev = u
```

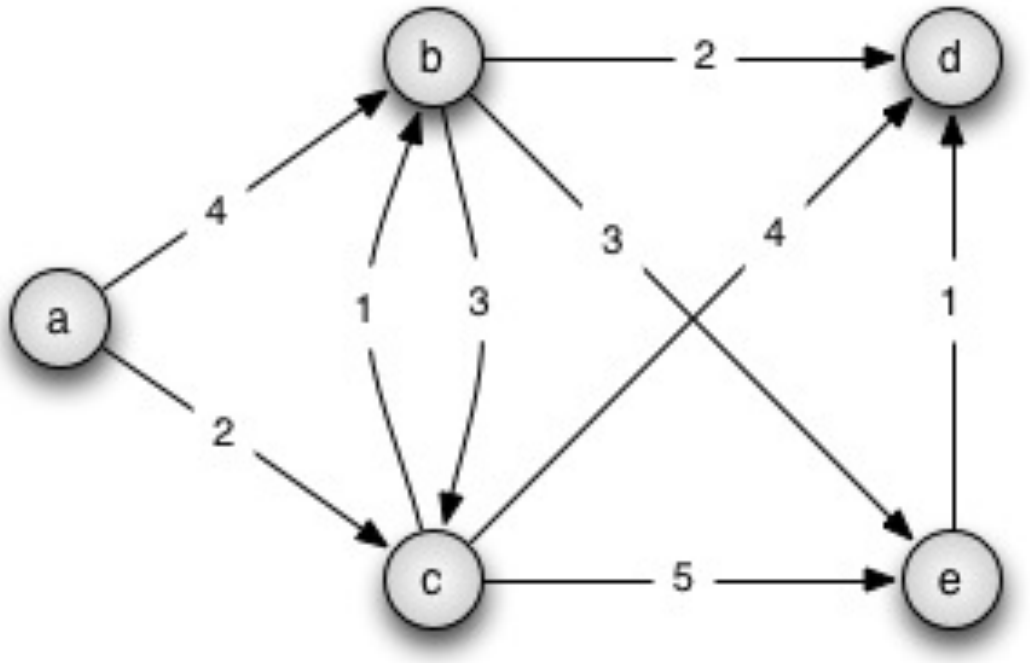
a bit more detail

```
input: graph G, weight function W, start node s

initialize:
all distances to ∞, except s.dist=0
set S=∅
priority queue Q containing all of V

while Q not empty
     u = Q.extractMin
     S = S ∪ {u}
     for each v ∈ adj[u]
          relax(u,v)  -- involves decreaseKey on Q
```

time just like Prim's

- depends on priority queue implementation
- set can be represented with a vector
- V inserts and extractMin's
- E decreaseKey's
- binary heap: $O((V+E) \lg V)$
- fibonacci heap: $O(V \lg V + E)$

example graph



greedy methods need greedy proof

- define $\delta(s,v)$ to be the the length of the shortest path from s to v
- … which may be different from v.dist, which is the shortest path found so far

one loop invariant:

at the start of each iteration of the while loop, $v.dist = \delta(s,v)$ for all $v \in S$

better loop invariant (can you see why?)

loop invariant: at the start of each iteration of the while loop

(i) for all $v \in S$, $v.dist = \delta(s,v)$

(ii) for all $v \notin S$, $v.dist$ is the length of the shortest path from s to v, *all of whose intermediate vertices are in S*

basic fact

If u is an intermediate vertex on the shortest path from s to v, then that part of the path from s to u is the shortest path to u.

In this context (no negative edge weights)

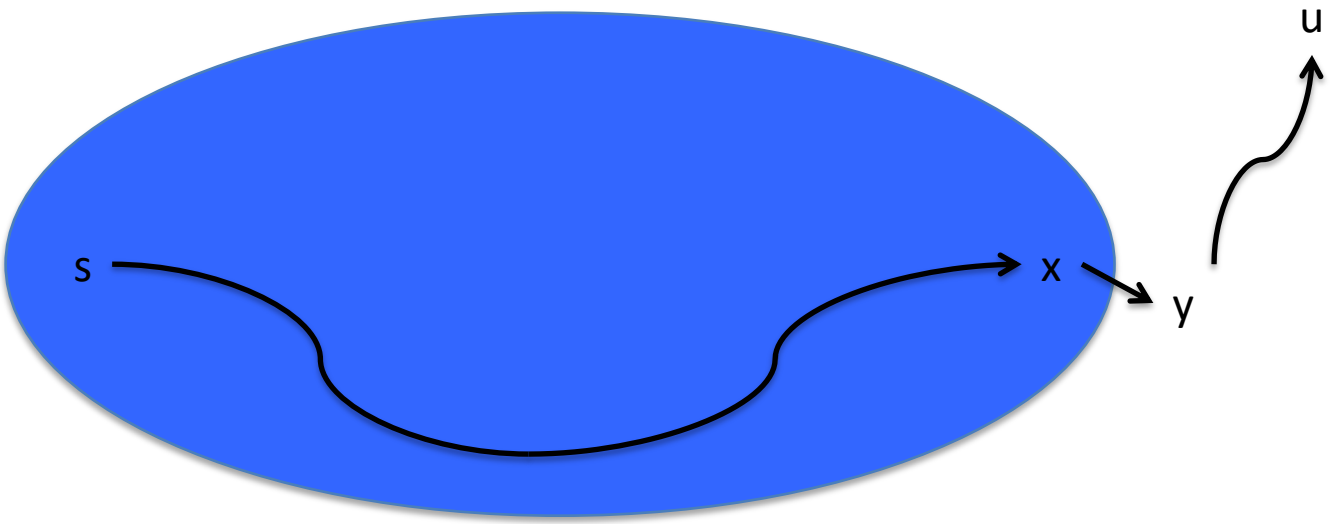
$\delta(s,u) < \delta(s,v)$

correctness using that invariant

- assume the invariant (parts (i) and (ii)) at the beginning of the loop
- let u be the chosen vertex with minimum u.dist
- we proceed by contradiction ….
- assume that u.dist is not the shortest path, that is, $\delta(s,u) < u.dist$

- continuing, with $\delta(s,u) < u.dist$
- part (ii) of invariant says that u.dist is the shortest path to u with intermediate vertices in S
- so the actual shortest path to u includes vertices not in S
- let y be the first vertex on that path not in S
- by the basic fact, that is the shortest path to y
- since intermediate vertices to y are in S, part (ii) of the loop invariant gives $\delta(s,y) = y.dist$

the situation



S (the set, in blue)

curved line is path
straight line is edge
y is first node outside set S

punch line:
$y.dist = \delta(s,y) < \delta(s,u) < u.dist$

concluding correctness

- since y.dist=δ(s,y)<δ(s,u)<u.dist, u would not have been the vertex chosen
- so by contradiction, if u was chosen then

$$\delta(s,u) = u.dist$$

- to prove part (ii) we use part (i) and the correctness of the relax method (skipped here)