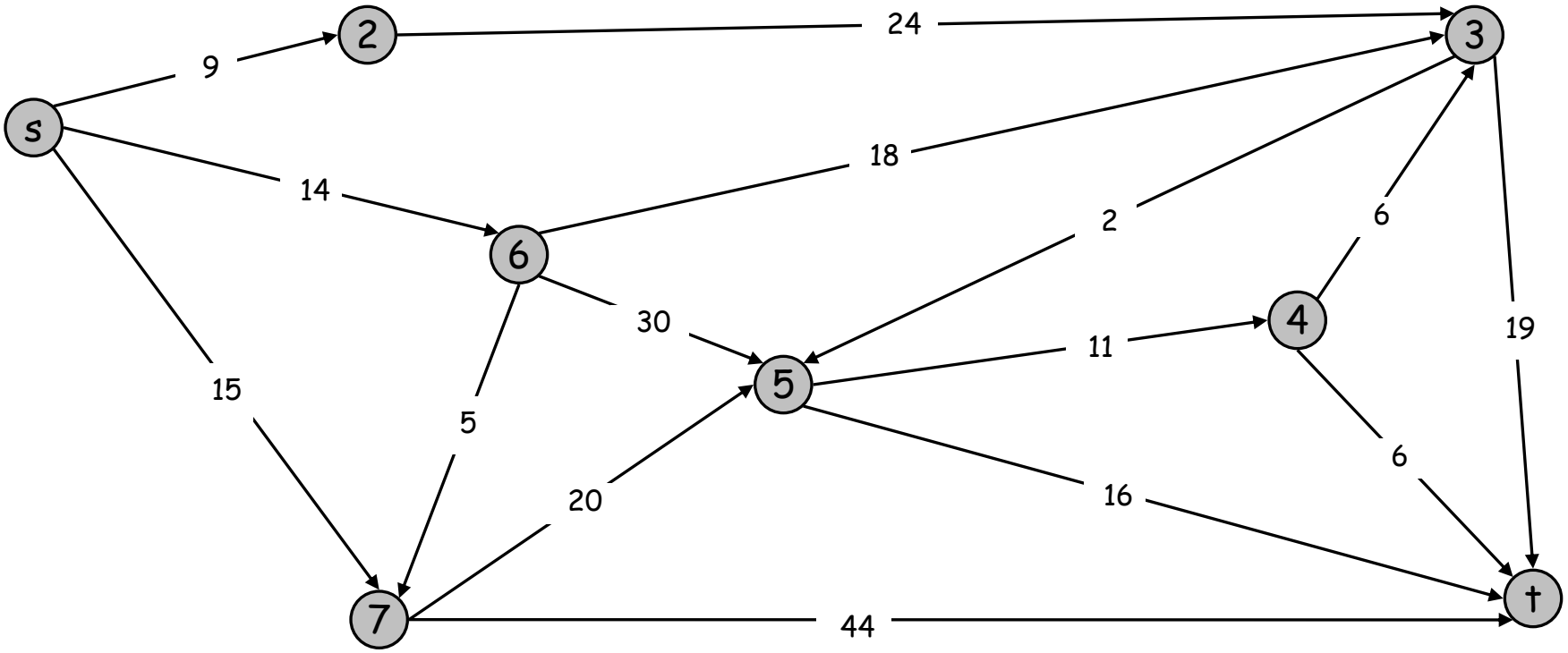
Lecture-5

DIJIKSTRA ALGORITHIM

Solution to the single-source shortest path problem in graph theory

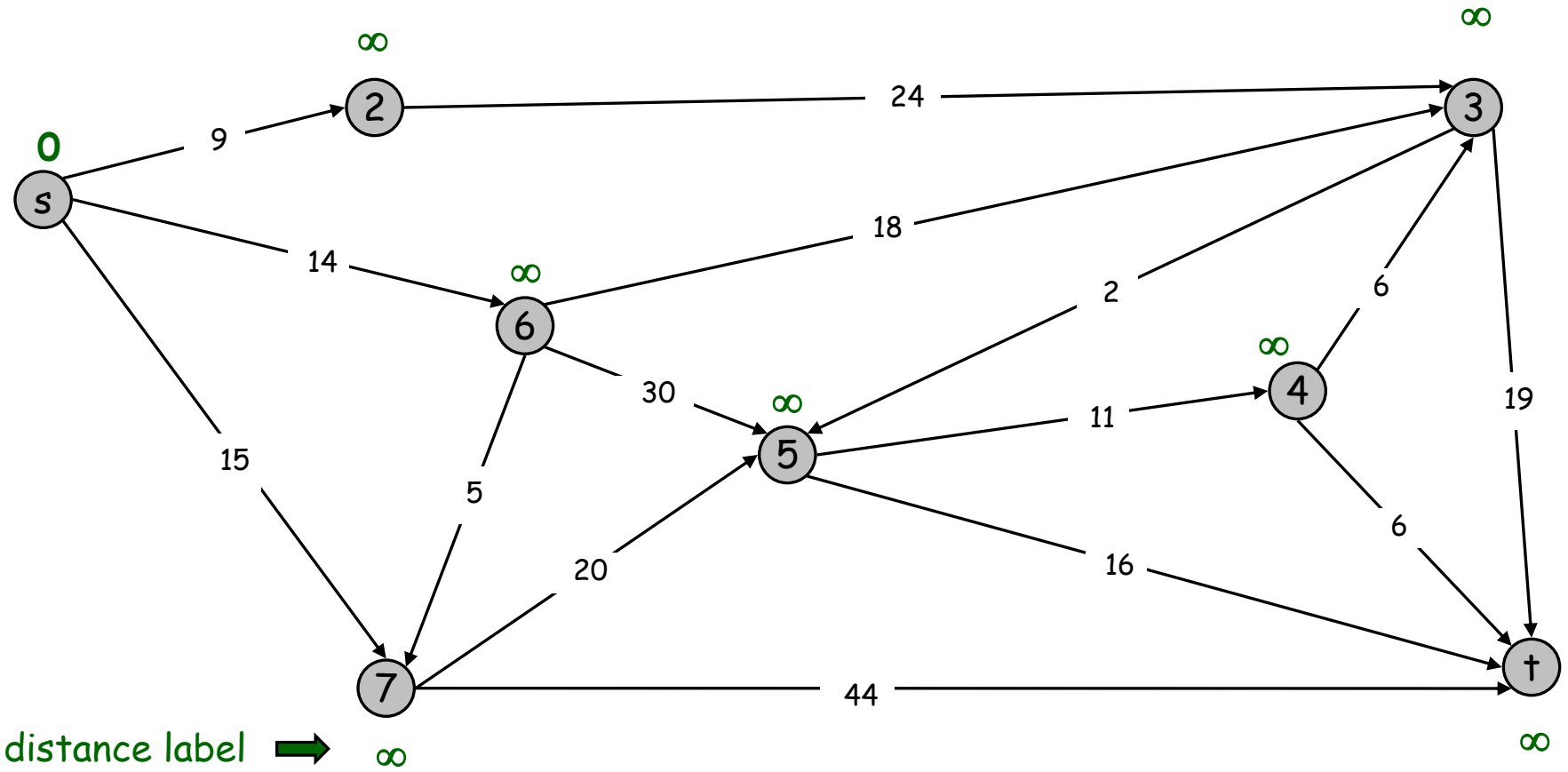
Dijkstras Shortest Path Algorithm

Find shortest path from s to t.



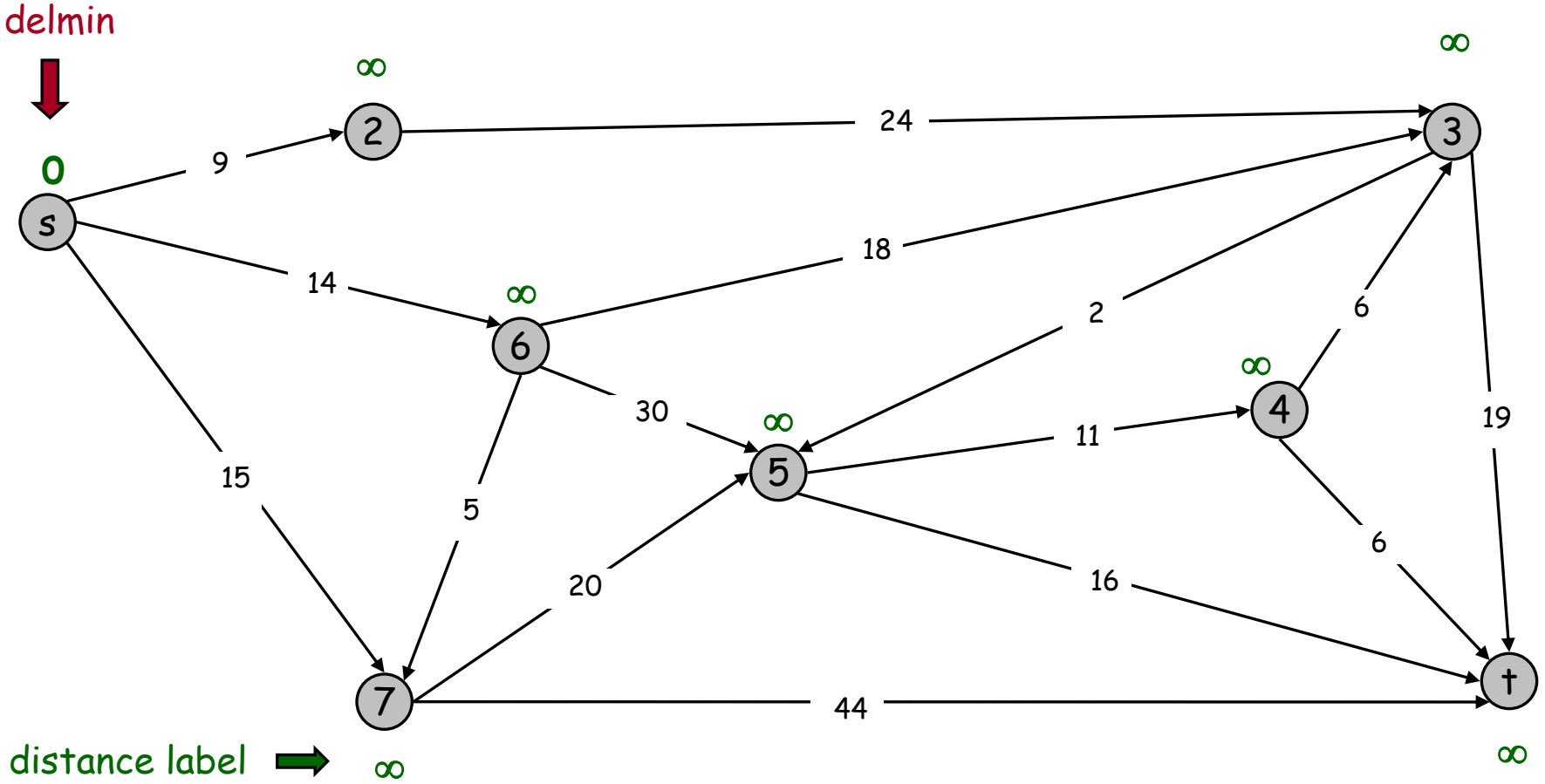
Dijkstras Shortest Path Algorithm

S = { }
PQ = { s, 2, 3, 4, 5, 6, 7, t }



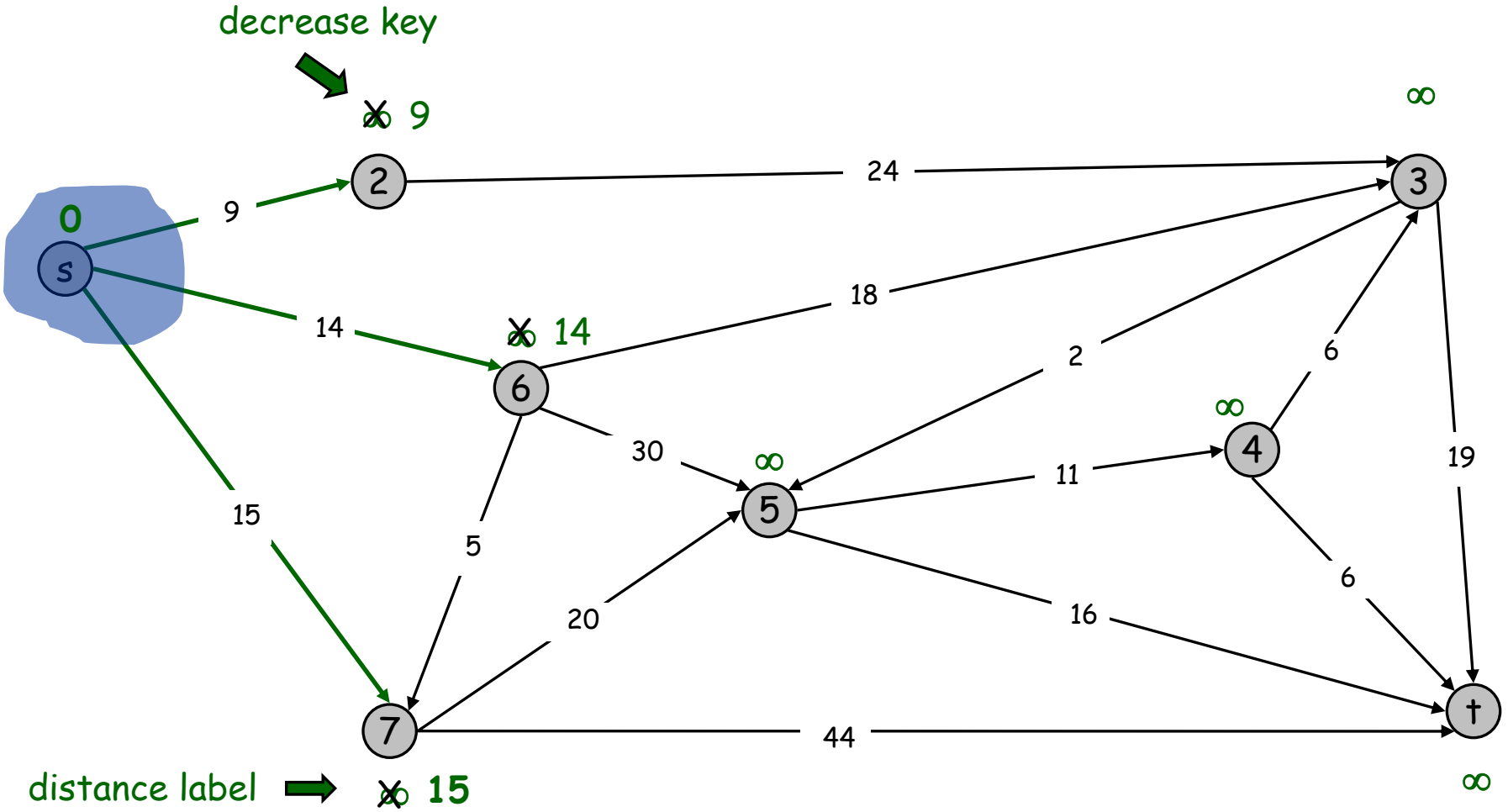
Dijkstras Shortest Path Algorithm

S = { }
PQ = { s, 2, 3, 4, 5, 6, 7, t }



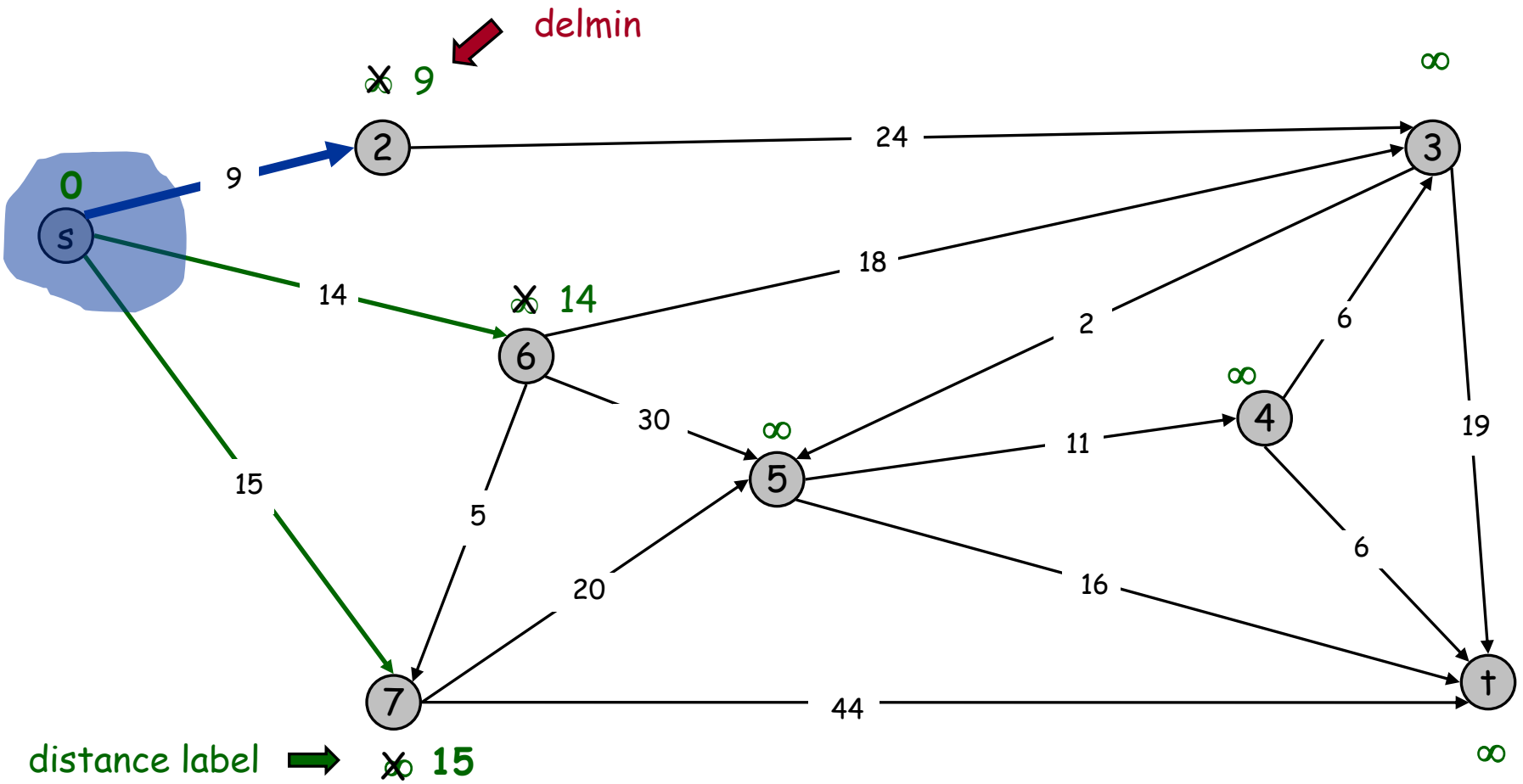
Dijkstras Shortest Path Algorithm

S = { s }
PQ = { 2, 3, 4, 5, 6, 7, t }



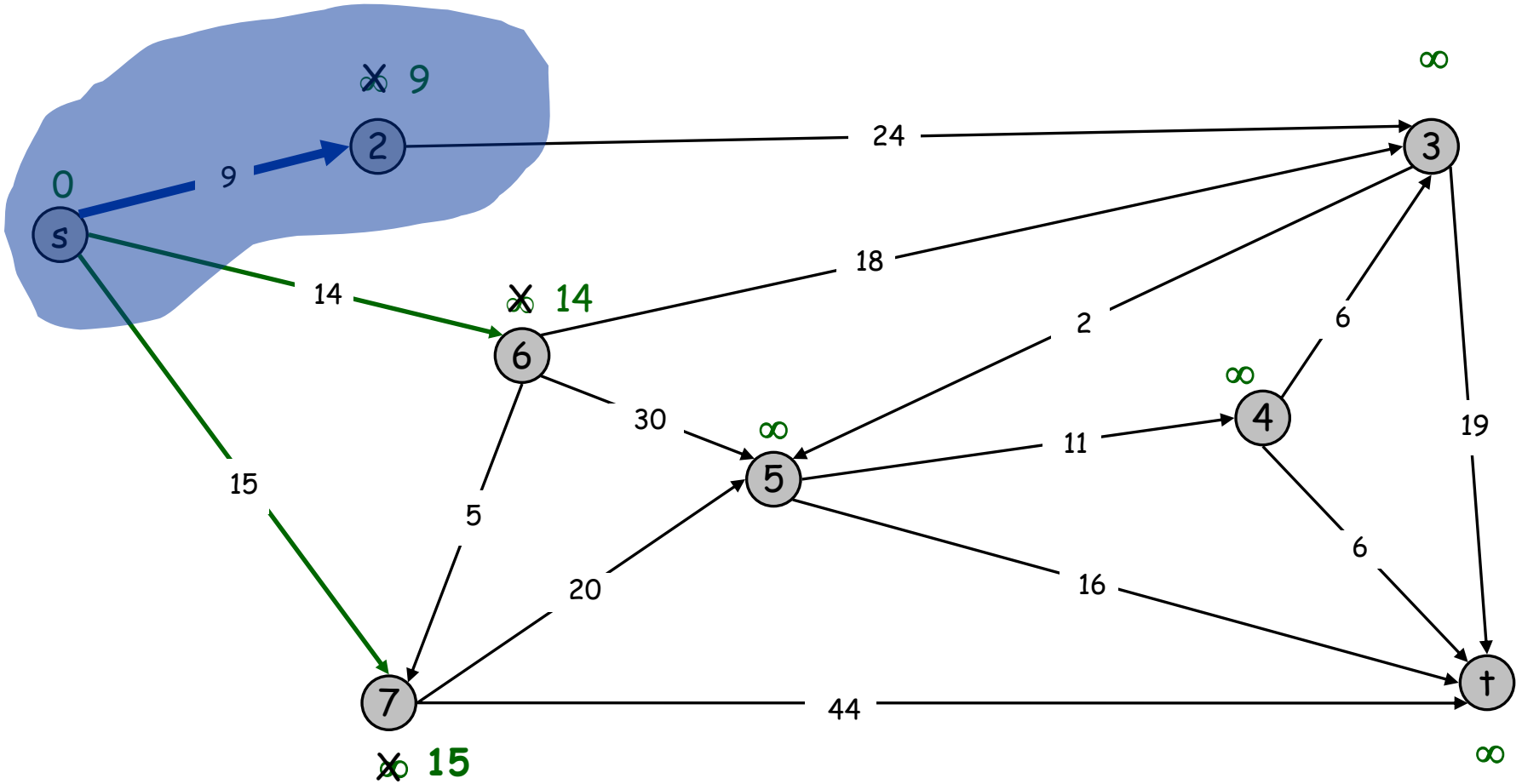
Dijkstras Shortest Path Algorithm

S = { s }
PQ = { 2, 3, 4, 5, 6, 7, t }



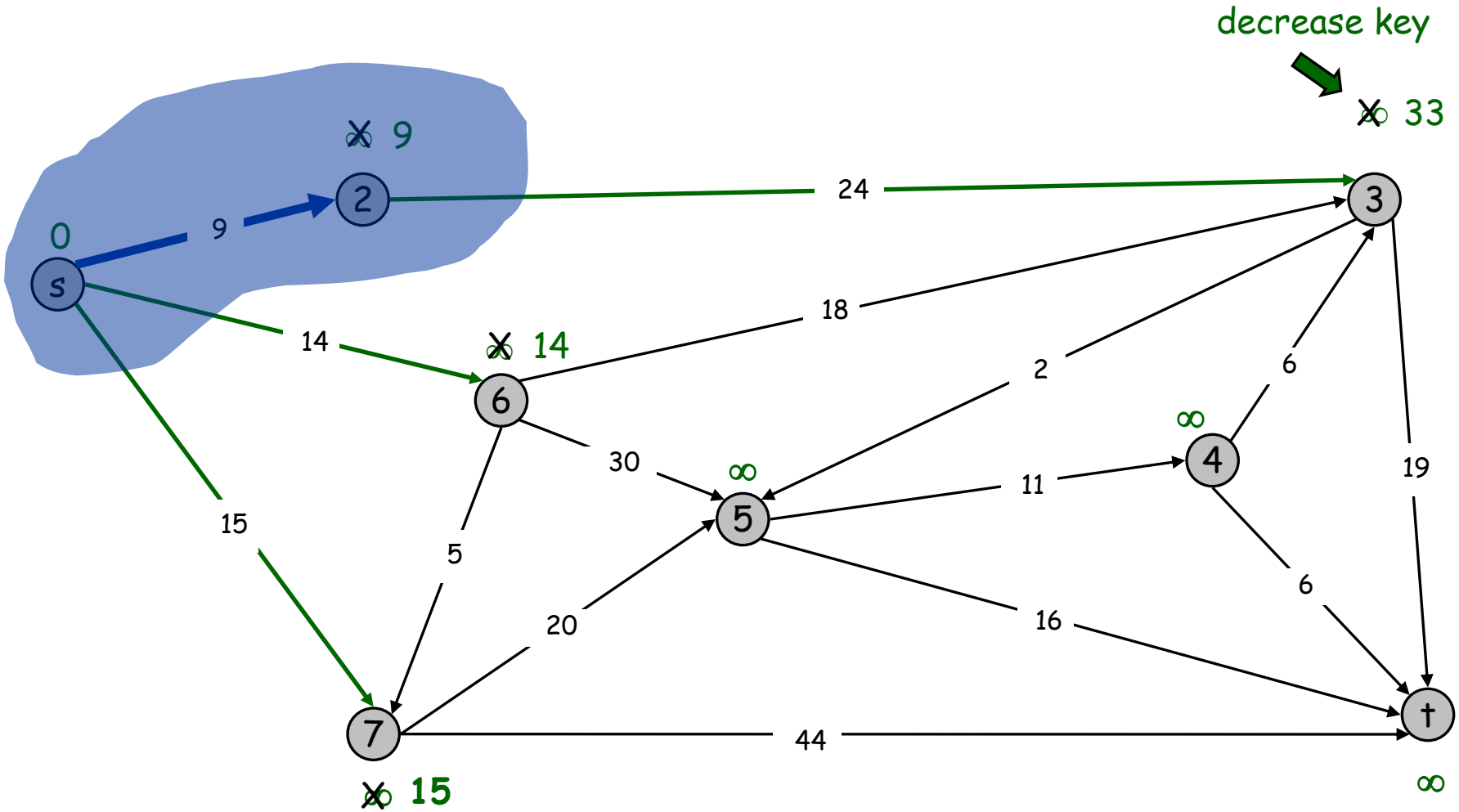
Dijkstras Shortest Path Algorithm

S = { s, 2 }
PQ = { 3, 4, 5, 6, 7, t }



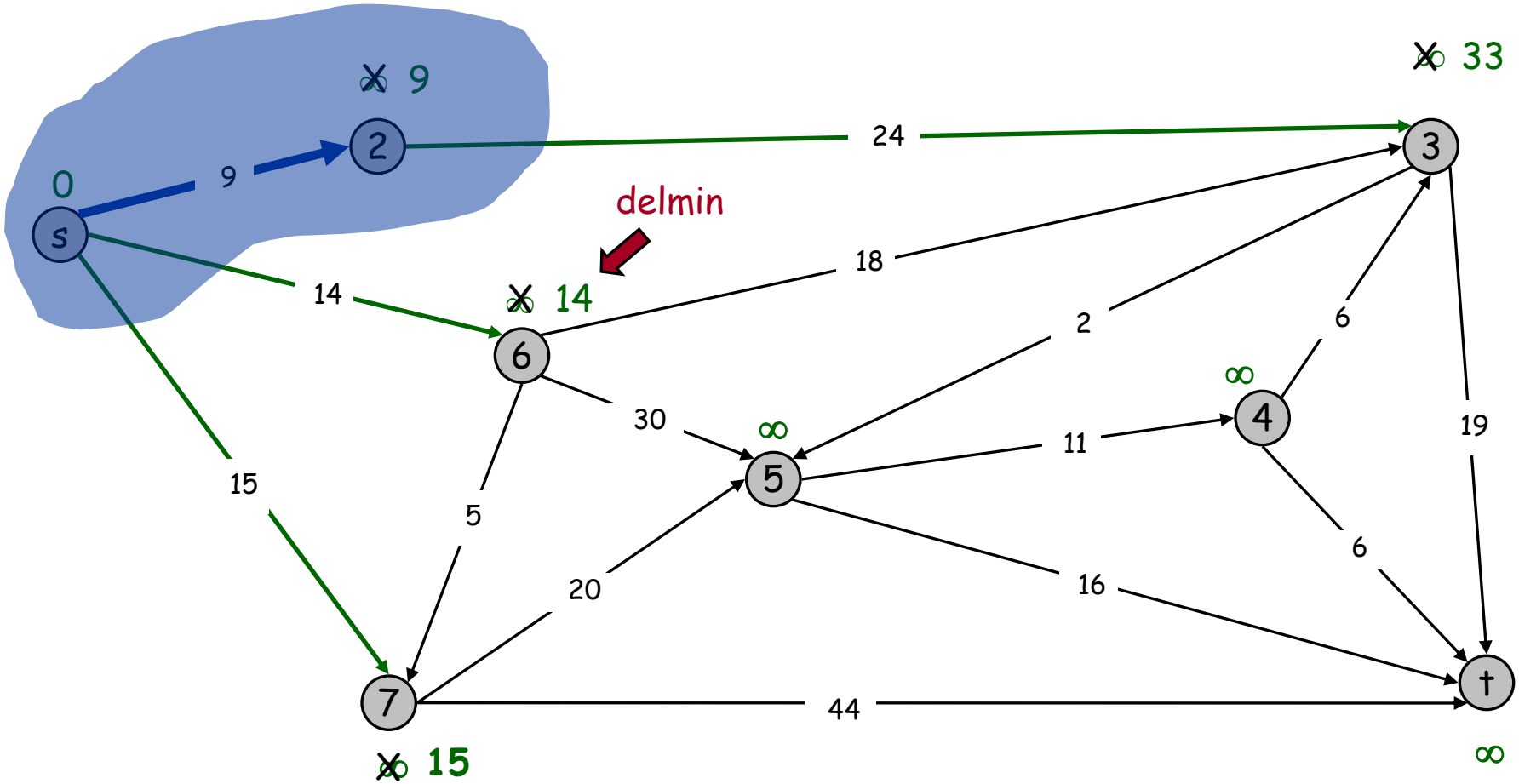
Dijkstras Shortest Path Algorithm

S = { s, 2 }
PQ = { 3, 4, 5, 6, 7, t }



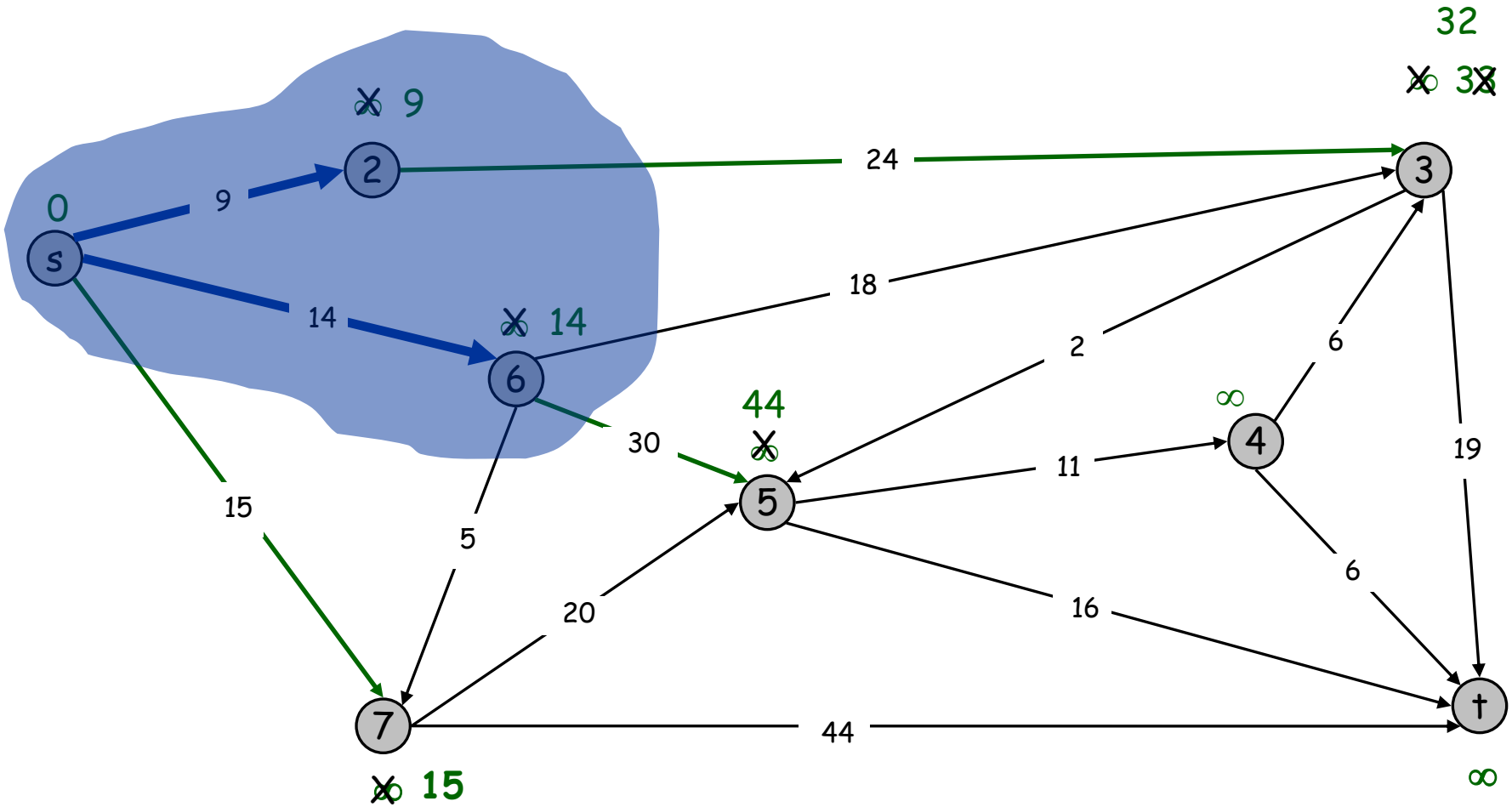
Dijkstras Shortest Path Algorithm

S = { s, 2 }
PQ = { 3, 4, 5, 6, 7, t }



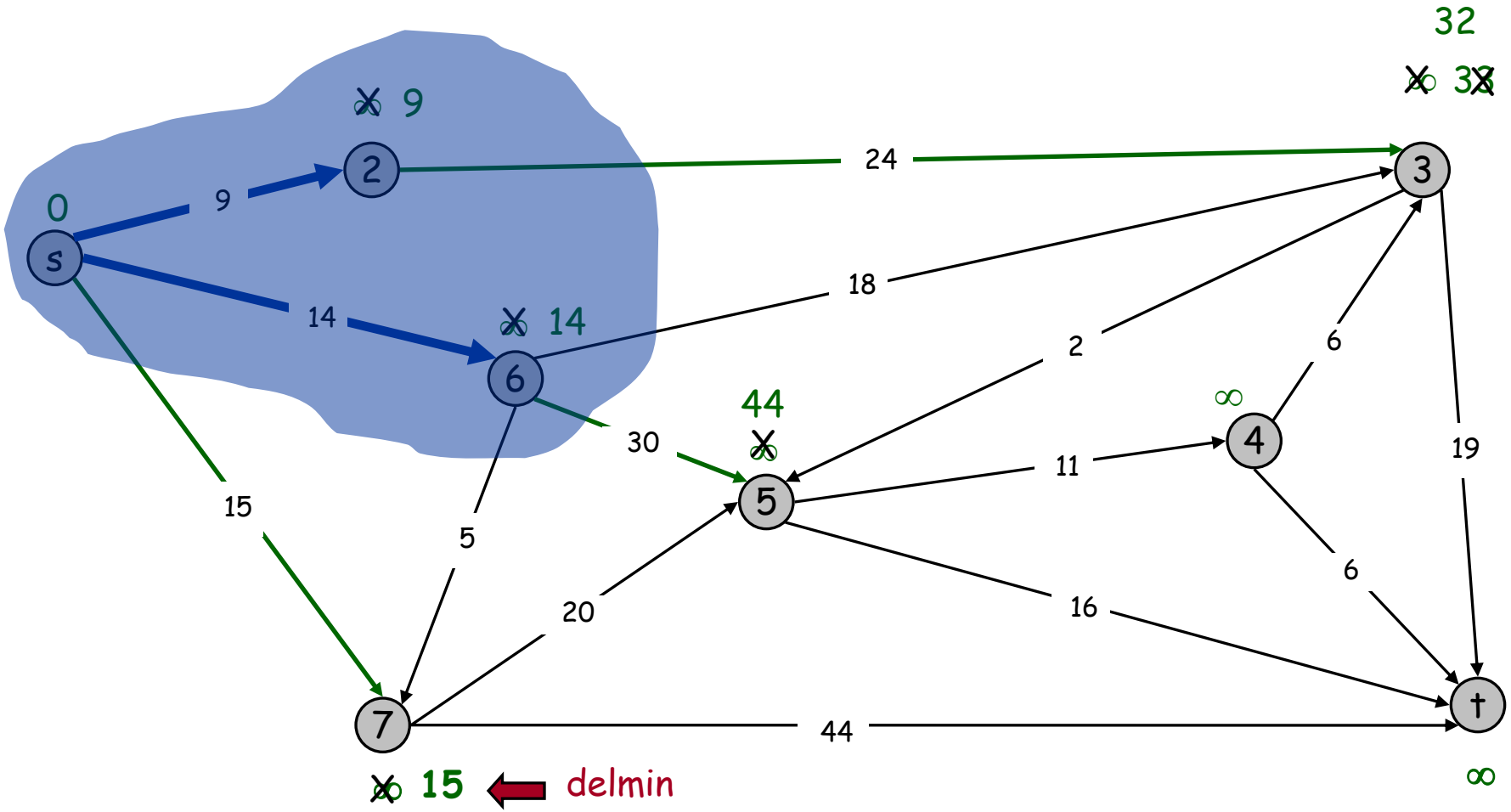
Dijkstras Shortest Path Algorithm

S = { s, 2, 6 }
PQ = { 3, 4, 5, 7, t }



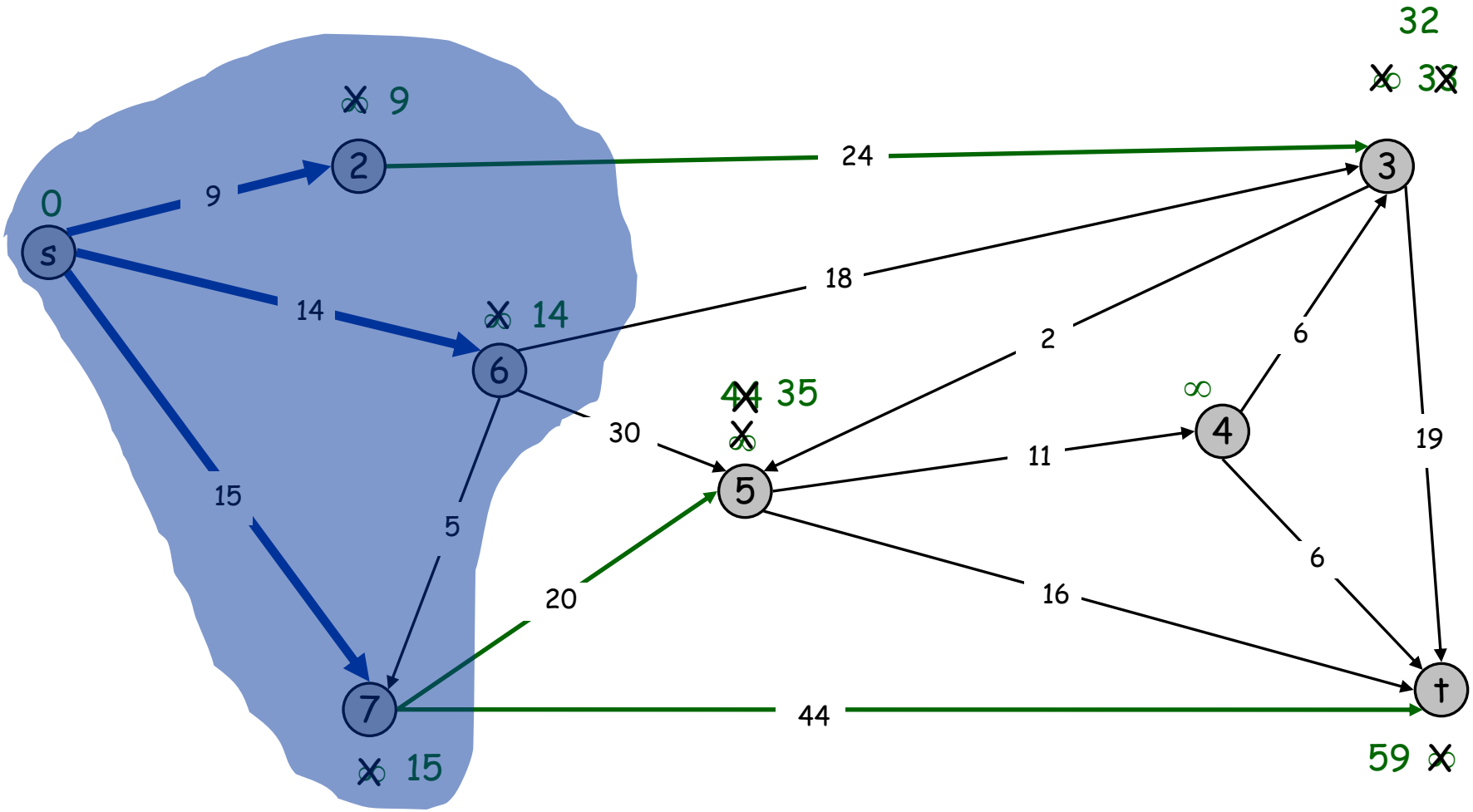
Dijkstras Shortest Path Algorithm

S = { s, 2, 6 }
PQ = { 3, 4, 5, 7, t }



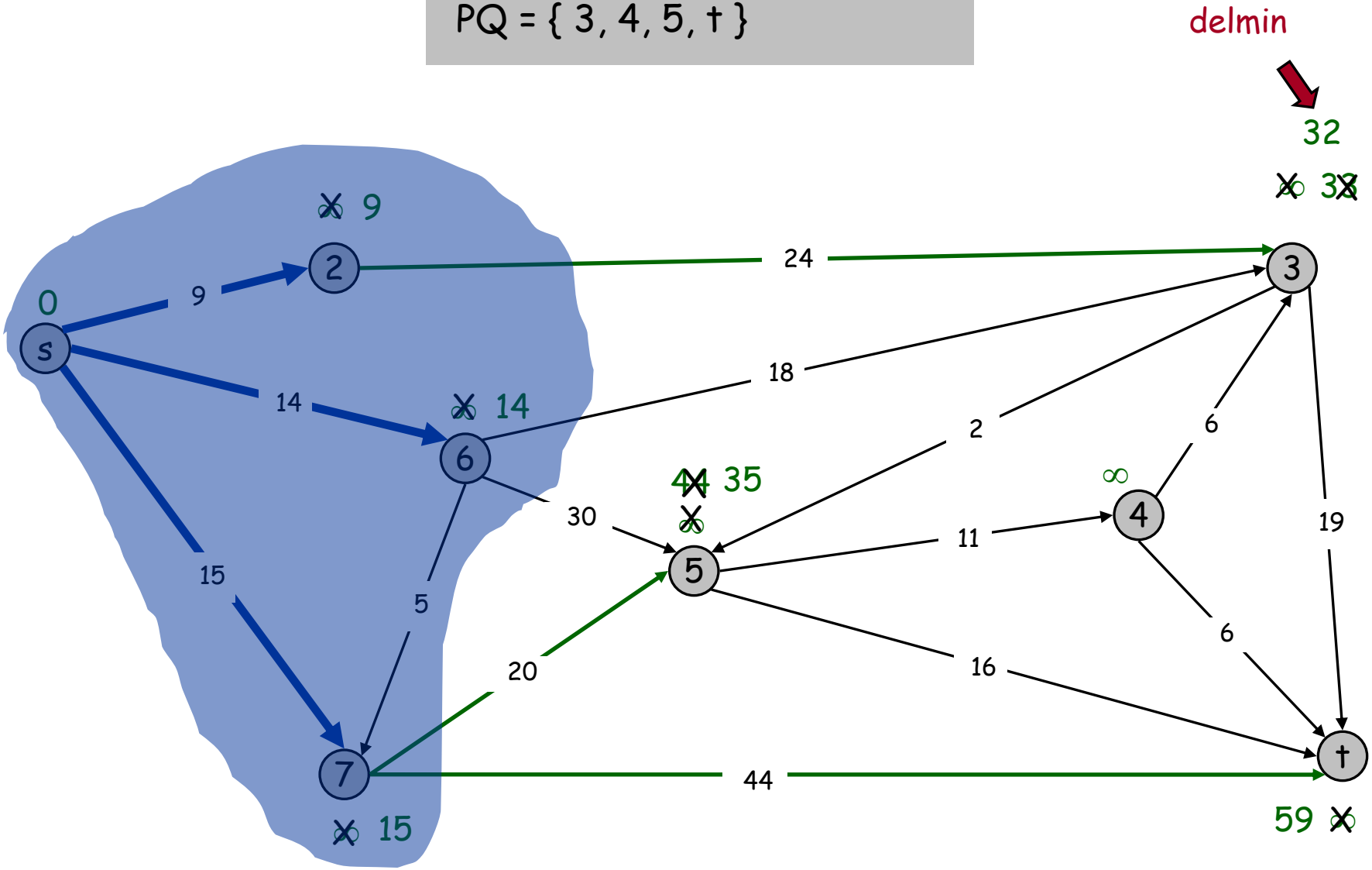
Dijkstras Shortest Path Algorithm

S = { s, 2, 6, 7 }
PQ = { 3, 4, 5, t }



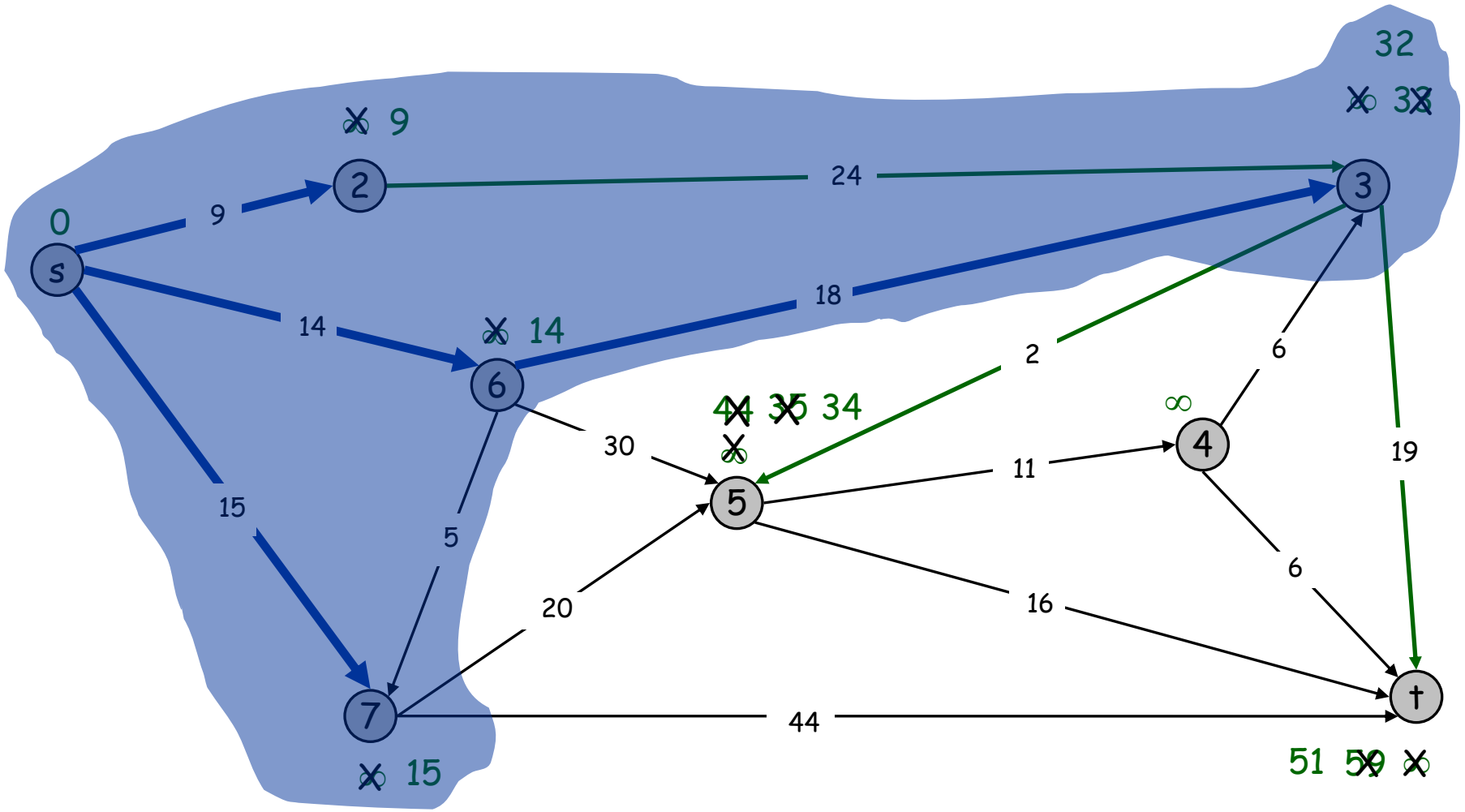
Dijkstras Shortest Path Algorithm

S = { s, 2, 6, 7 }
PQ = { 3, 4, 5, t }



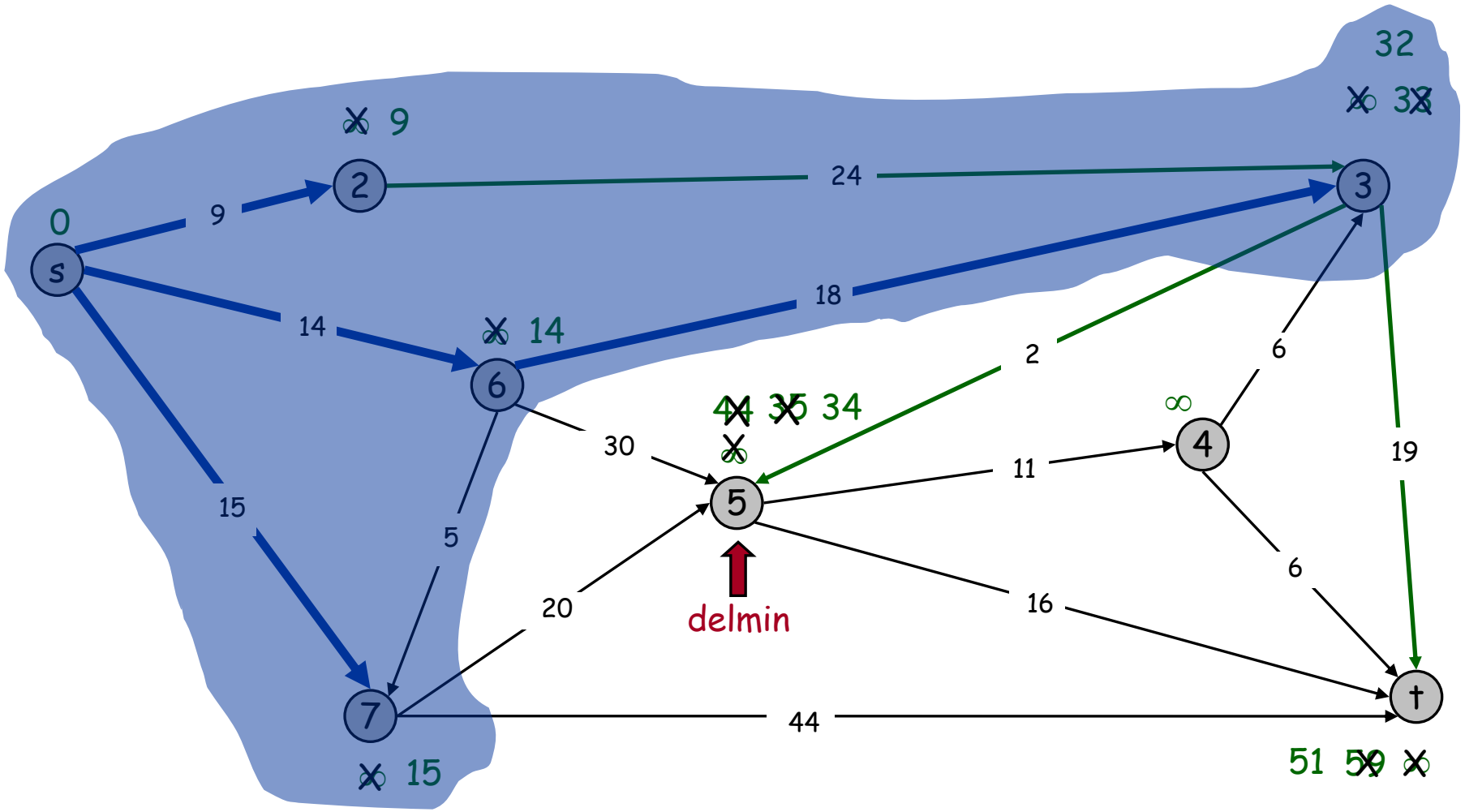
Dijkstras Shortest Path Algorithm

S = { s, 2, 3, 6, 7 }
PQ = { 4, 5, t }



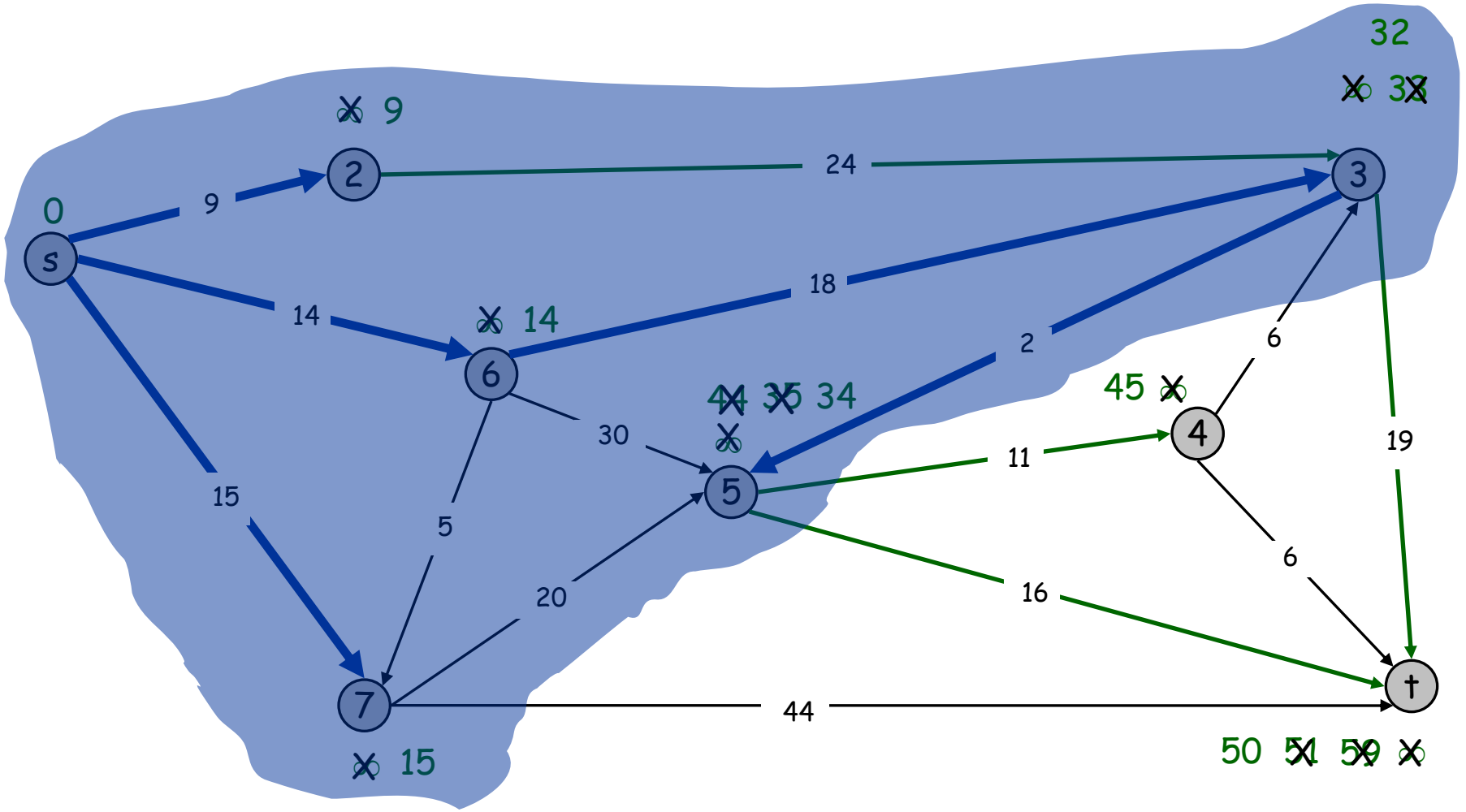
Dijkstras Shortest Path Algorithm

S = { s, 2, 3, 6, 7 }
PQ = { 4, 5, t }



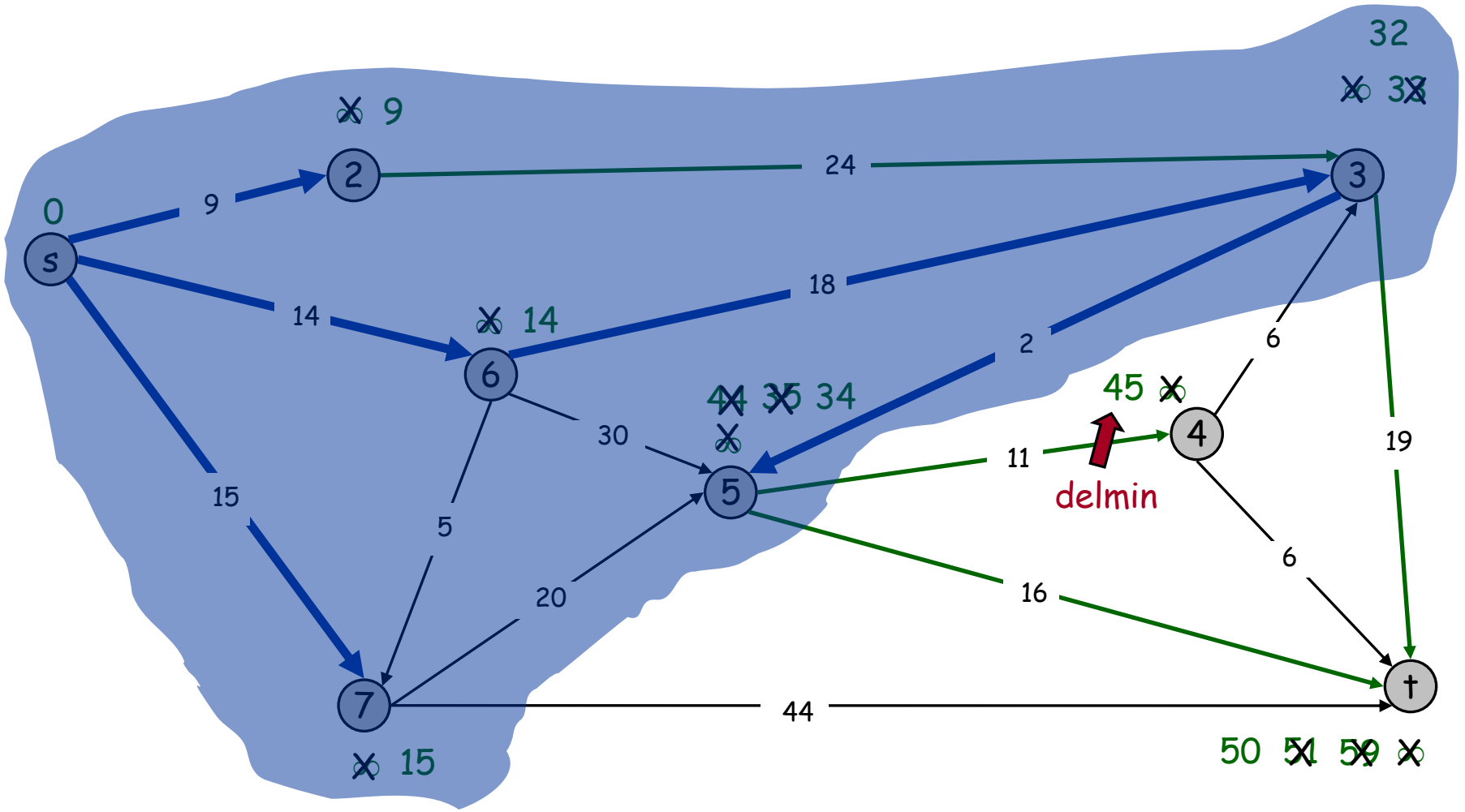
Dijkstras Shortest Path Algorithm

S = { s, 2, 3, 5, 6, 7 }
PQ = { 4, t }



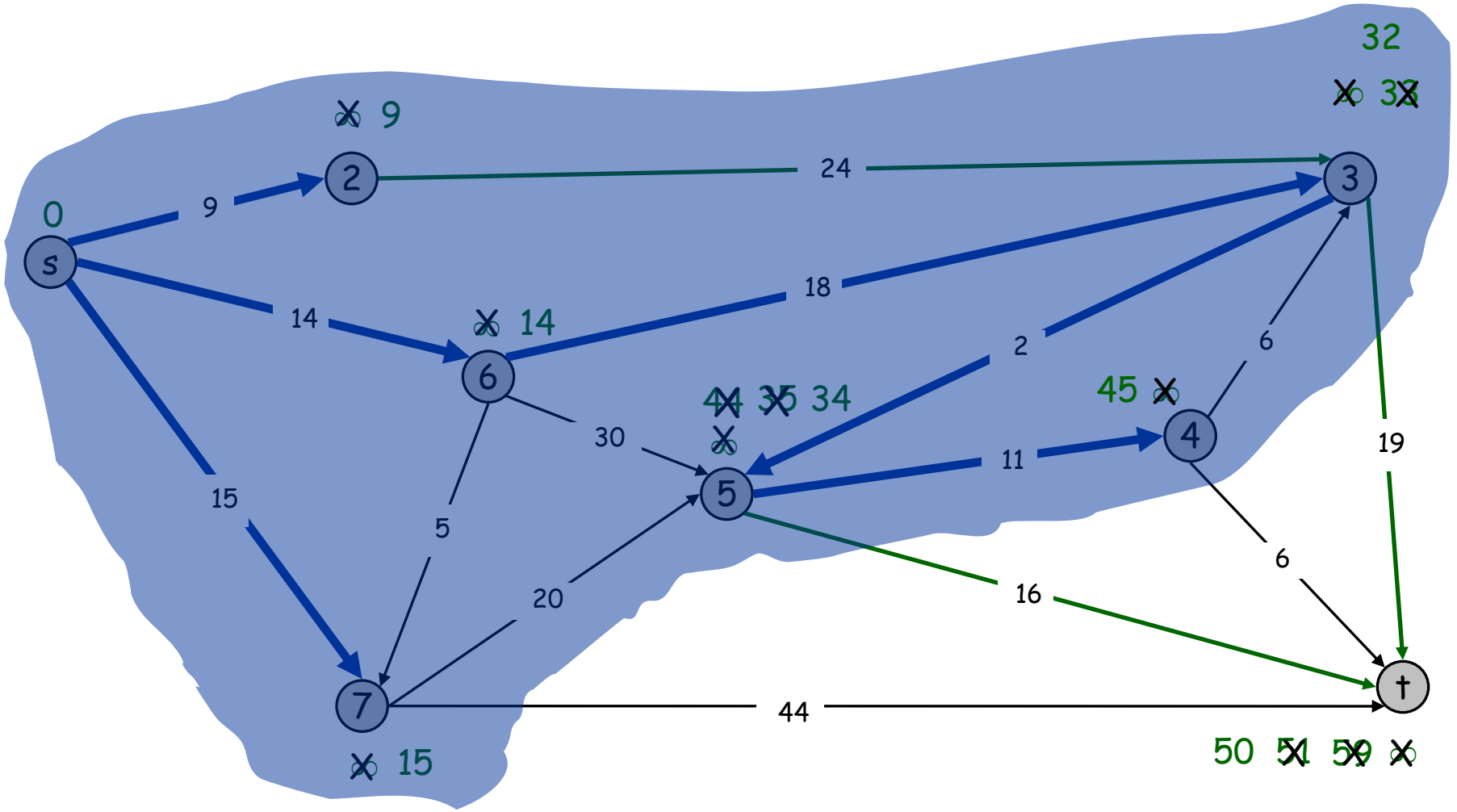
Dijkstras Shortest Path Algorithm

S = { s, 2, 3, 5, 6, 7 }
PQ = { 4, t }



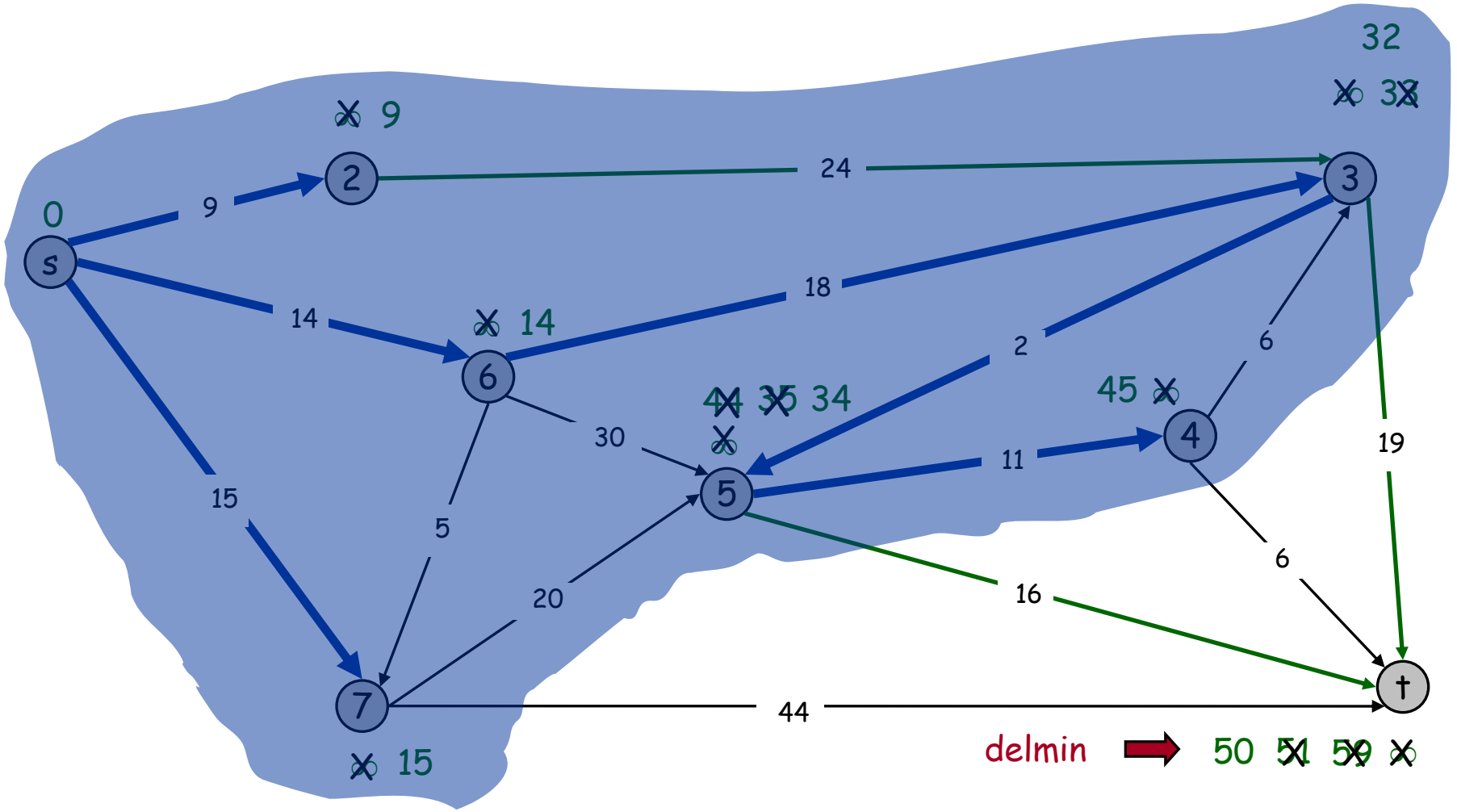
Dijkstras Shortest Path Algorithm

S = { s, 2, 3, 4, 5, 6, 7 }
PQ = { t }



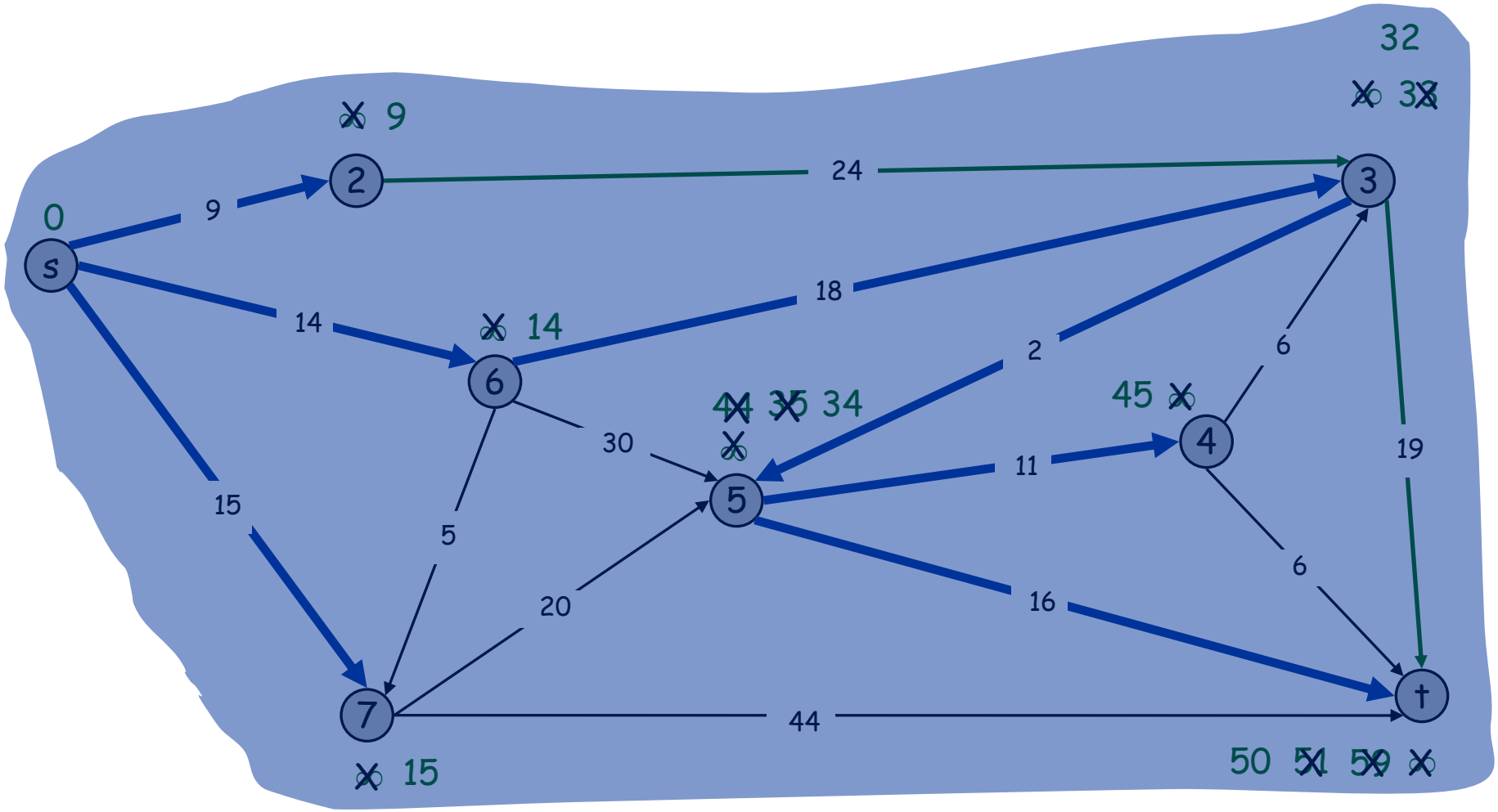
Dijkstras Shortest Path Algorithm

S = { s, 2, 3, 4, 5, 6, 7 }
PQ = { t }



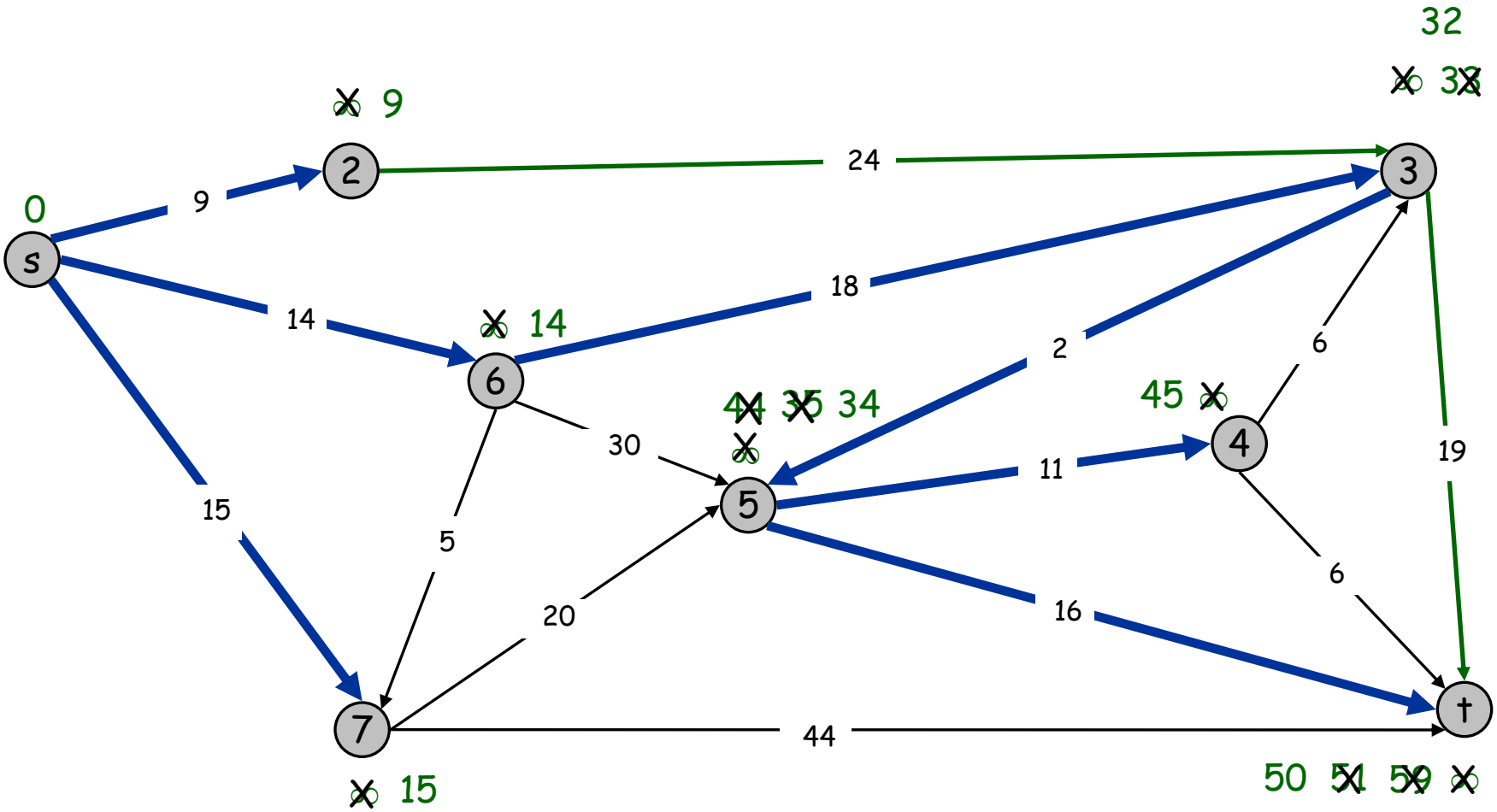
Dijkstras Shortest Path Algorithm

S = { s, 2, 3, 4, 5, 6, 7, t }
PQ = { }



Dijkstras Shortest Path Algorithm

S = { s, 2, 3, 4, 5, 6, 7, t }
PQ = { }



Why do we use Dijkstras Algorithm?

Algorithms for calculating shortest path from source to sink about as computationally expensive as calculating shortest paths from source to any vertex.

Pseudocode

Pseudocode

```
dist[s]←0                          (distancetosourcevertexiszero)
forallv∈V-{s}
 do dist[v]←∞                      (setallotherdistancestoinfinity)
S←∅                                (S,thesetofvisitedverticesisinitiallyempty)
Q←V                                (Q,thequeueinitiallycontainsallvertices)
whileQ≠∅                           (whilethequeueisnotempty)
dou← mindistance(Q,dist)           (selecttheelementofQwiththemin.distance)
S←S∪{u}
forallv∈neighbors[u]
doifdist[v]>dist[u]+w(u,v)         (ifnewshortestpathfound)
 thend[v]←d[u]+w(u,v)              (setnewvalueofshortestpath)
                                   (ifdesired,addtracebackcode)

Return dist
```