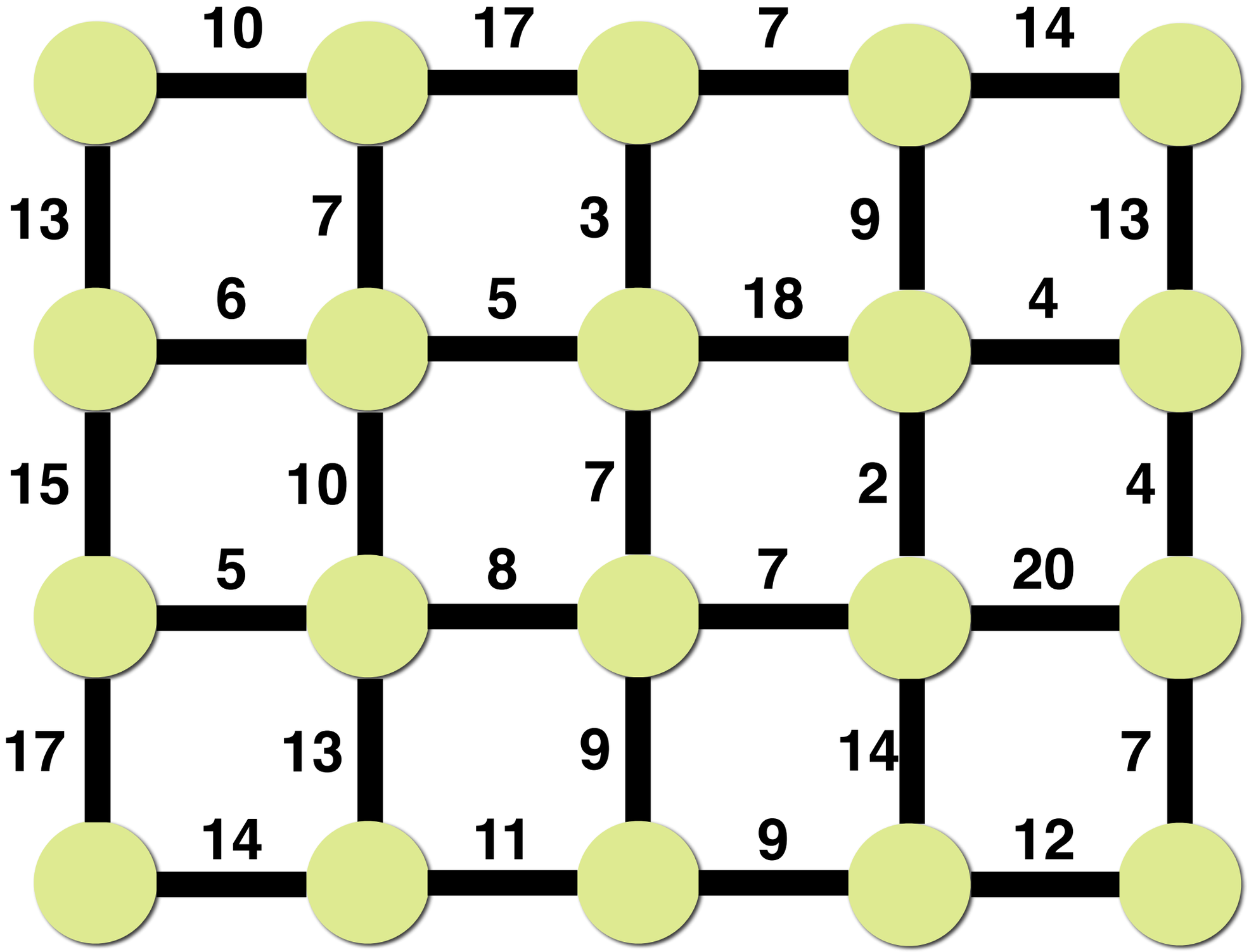
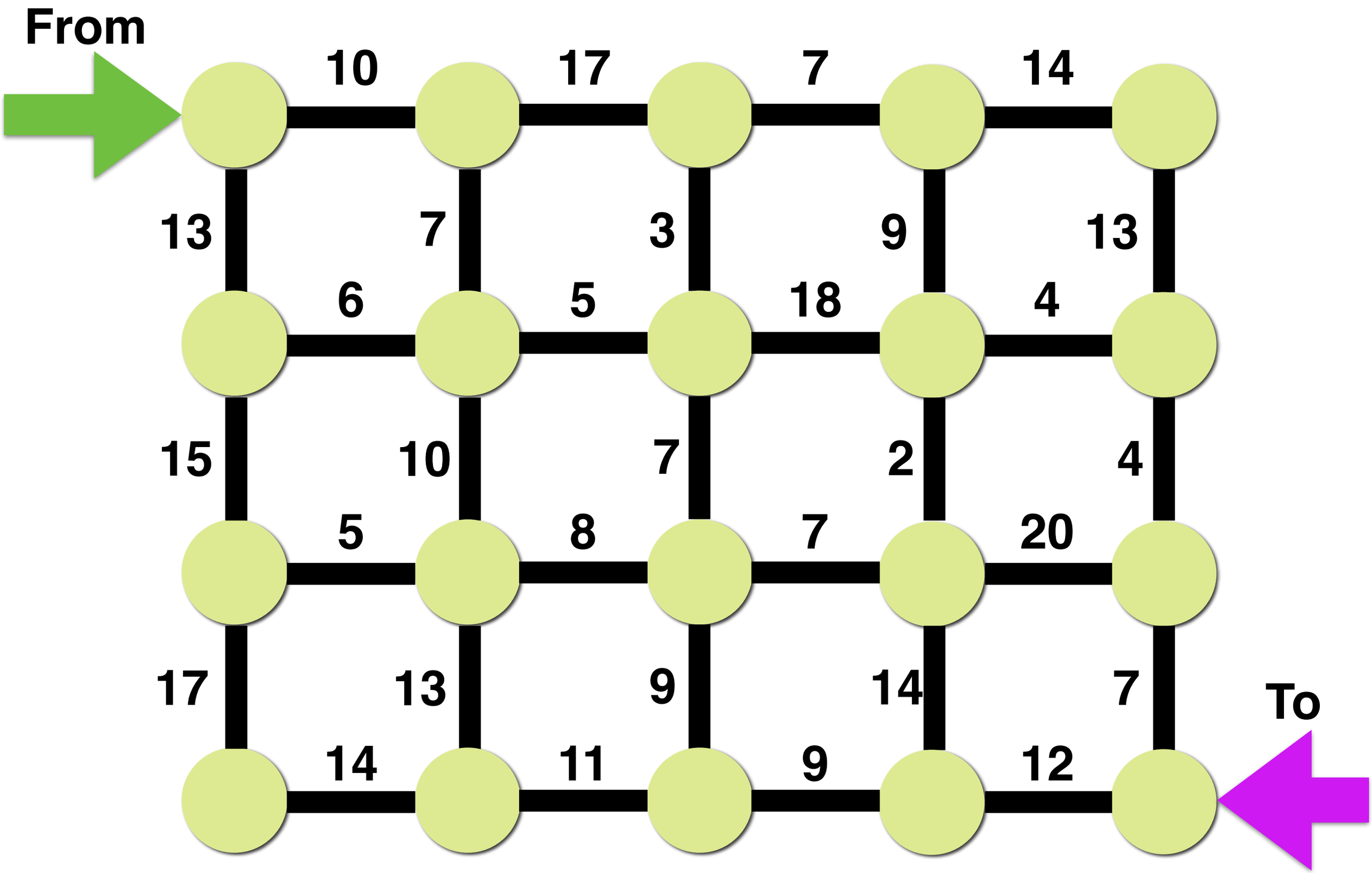
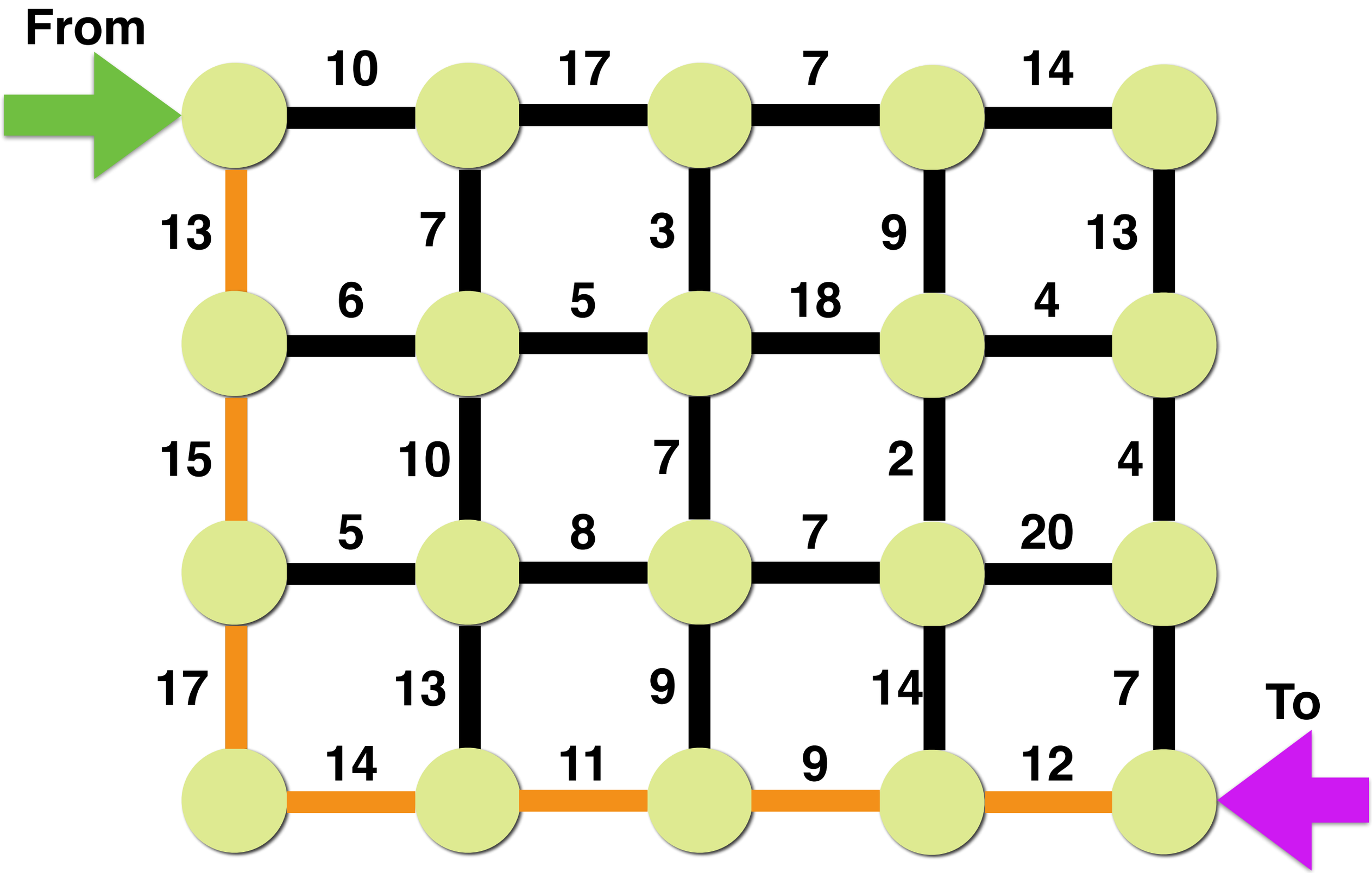
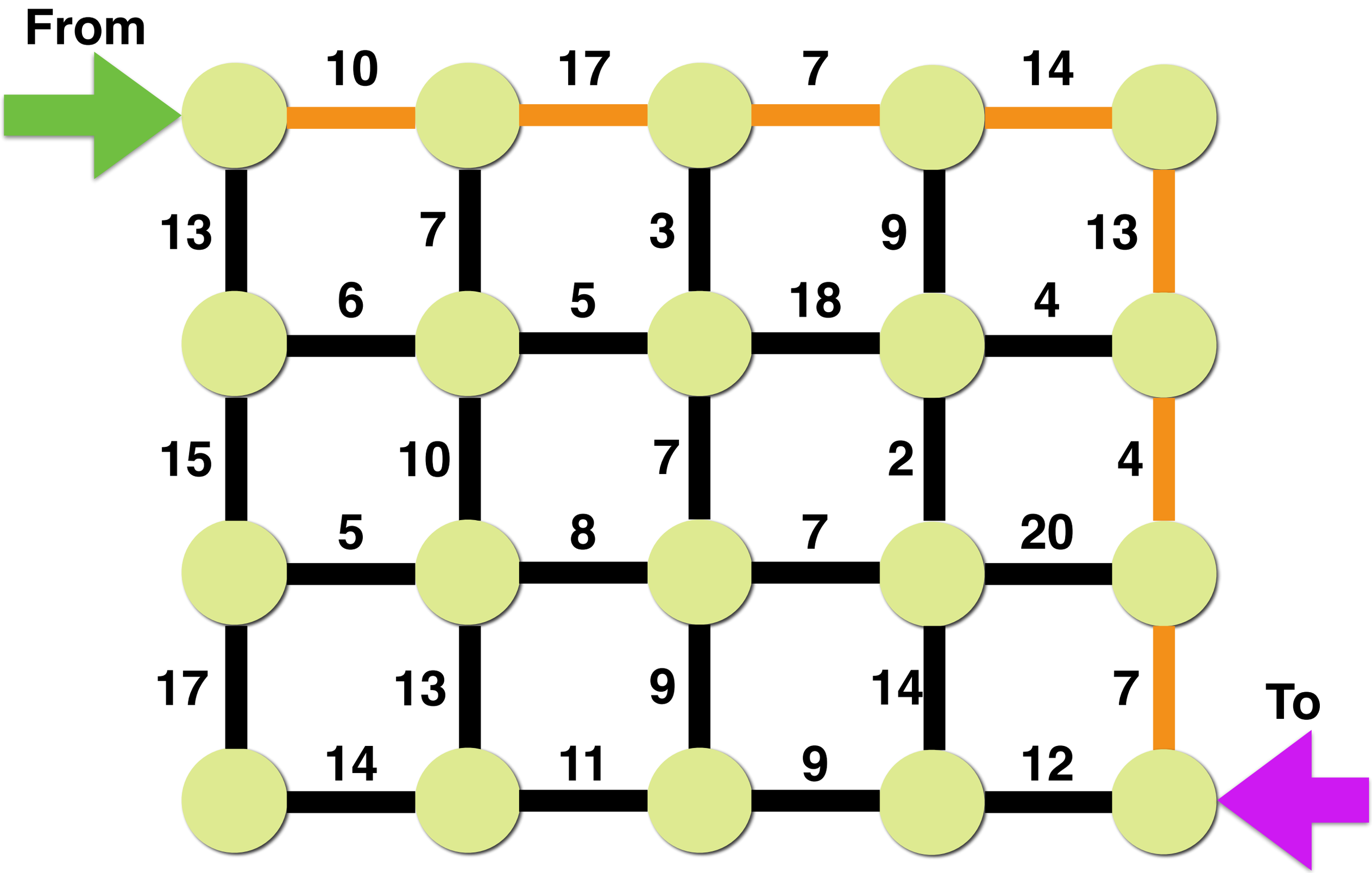
Single-Source Shortest Path problem


10
17
7
14
13
7
3
9
13
6
5
18
4
15
10
7
2
4
5
8
7
20
17
13
9
14
7
14
11
9
12

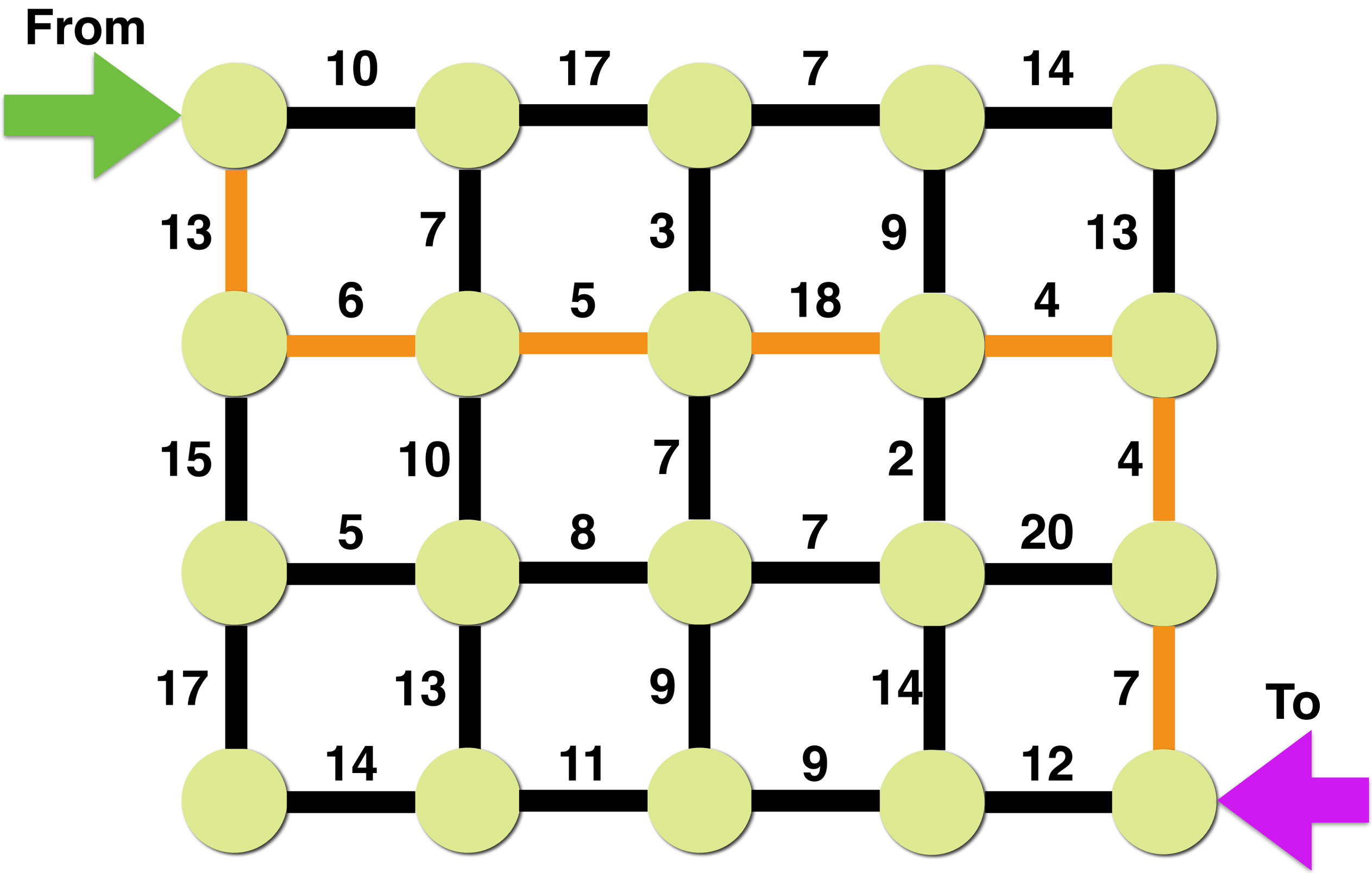

From
10
17
7
14
13
7
3
9
13
6
5
18
4
15
10
7
2
4
5
8
7
20
17
13
9
14
7
14
11
9
12
To



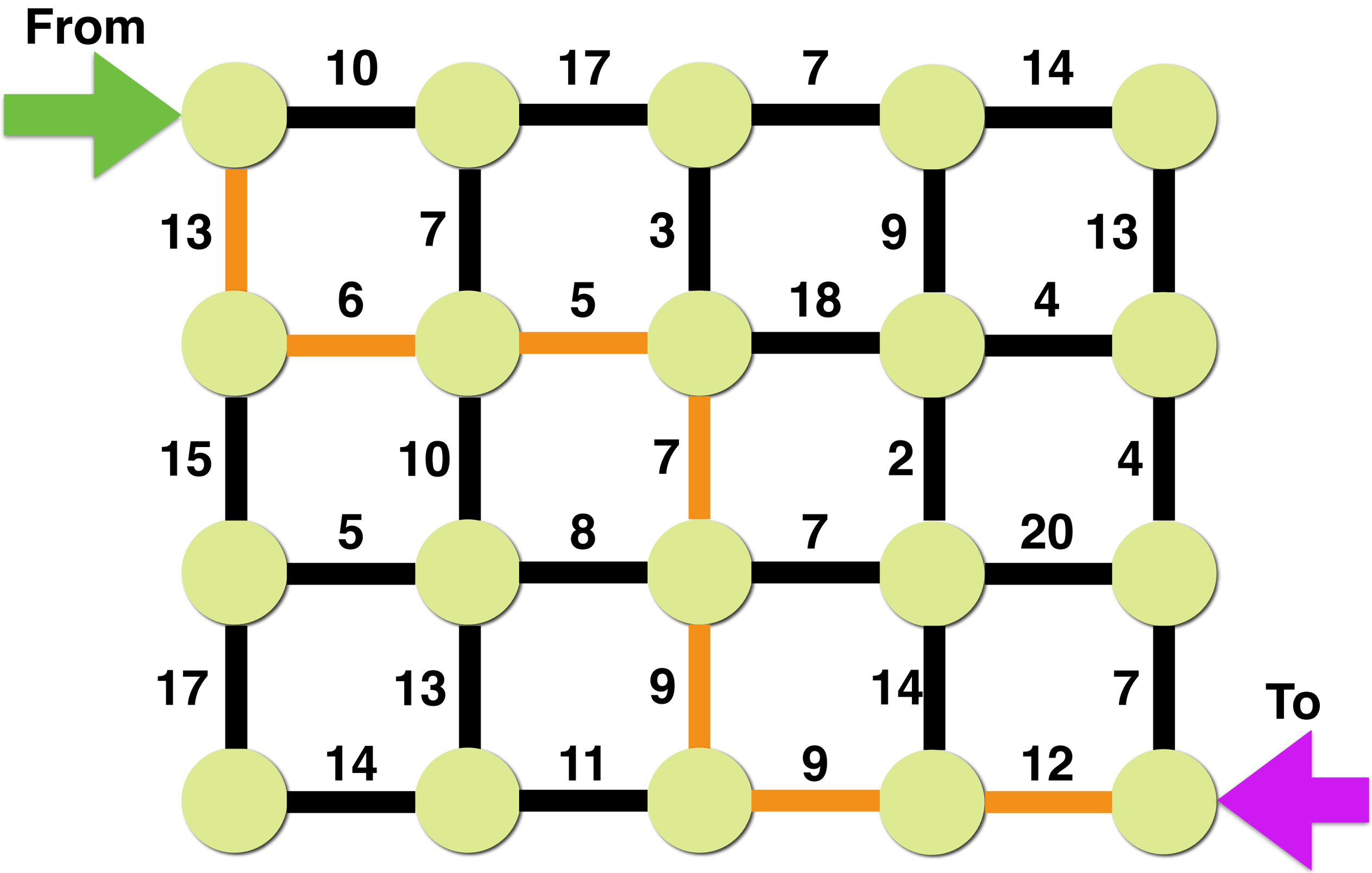
Travel Time: 13 + 15 + 17 + 14 + 11 + 9 + 12 = 91

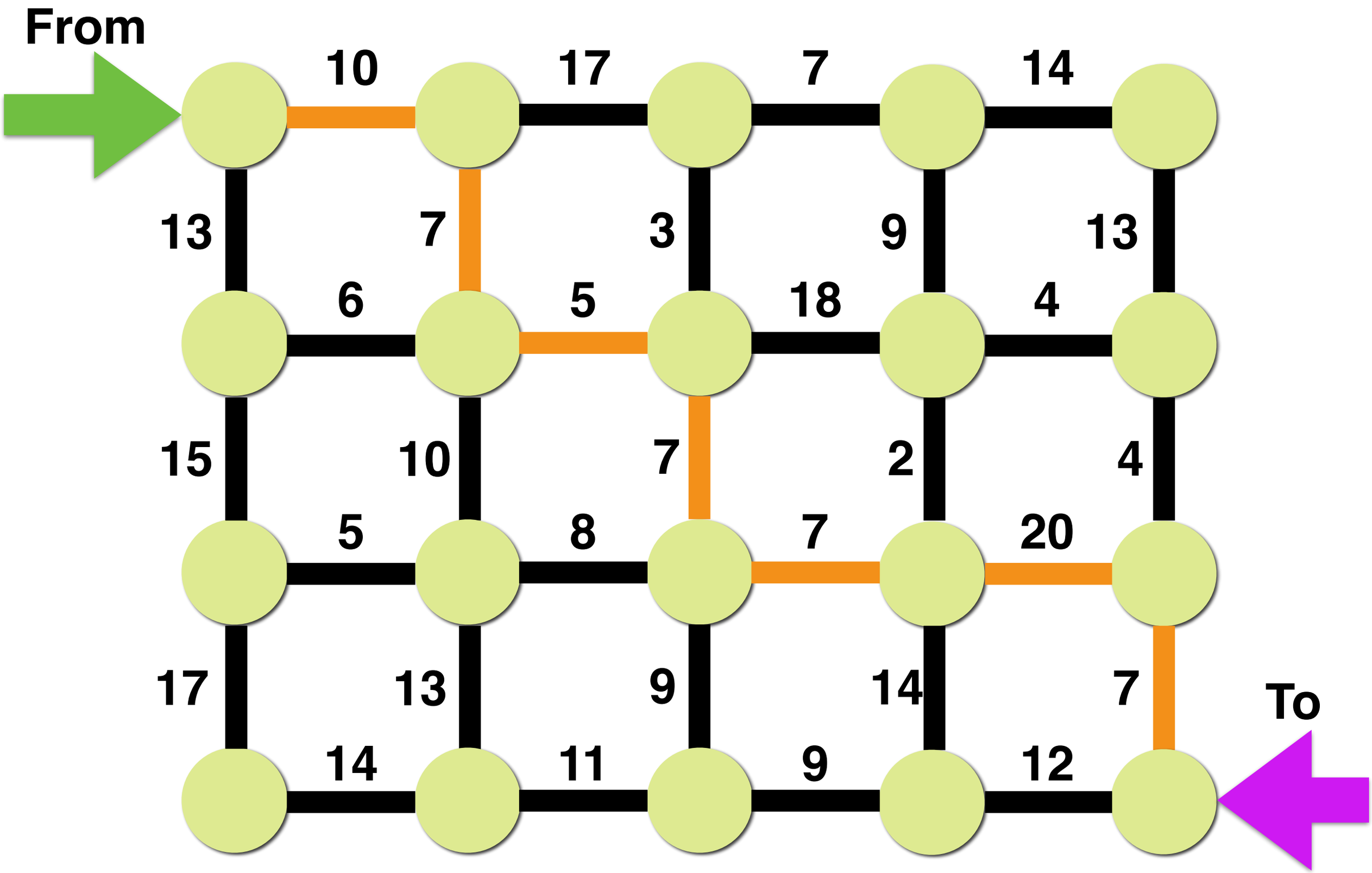


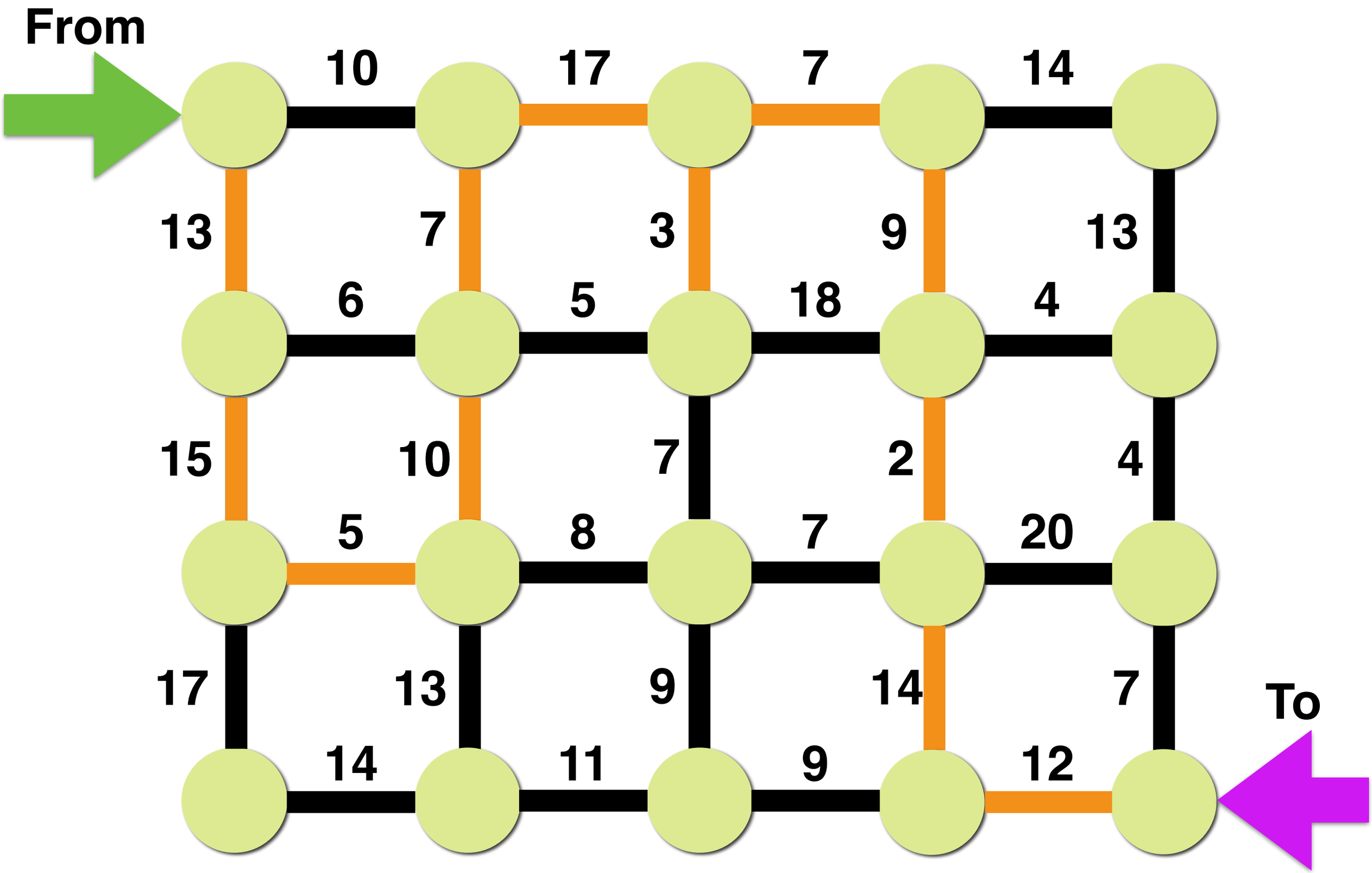
Travel Time: 10 + 17 + 7 + 14 + 13 + 4 + 7 = 72

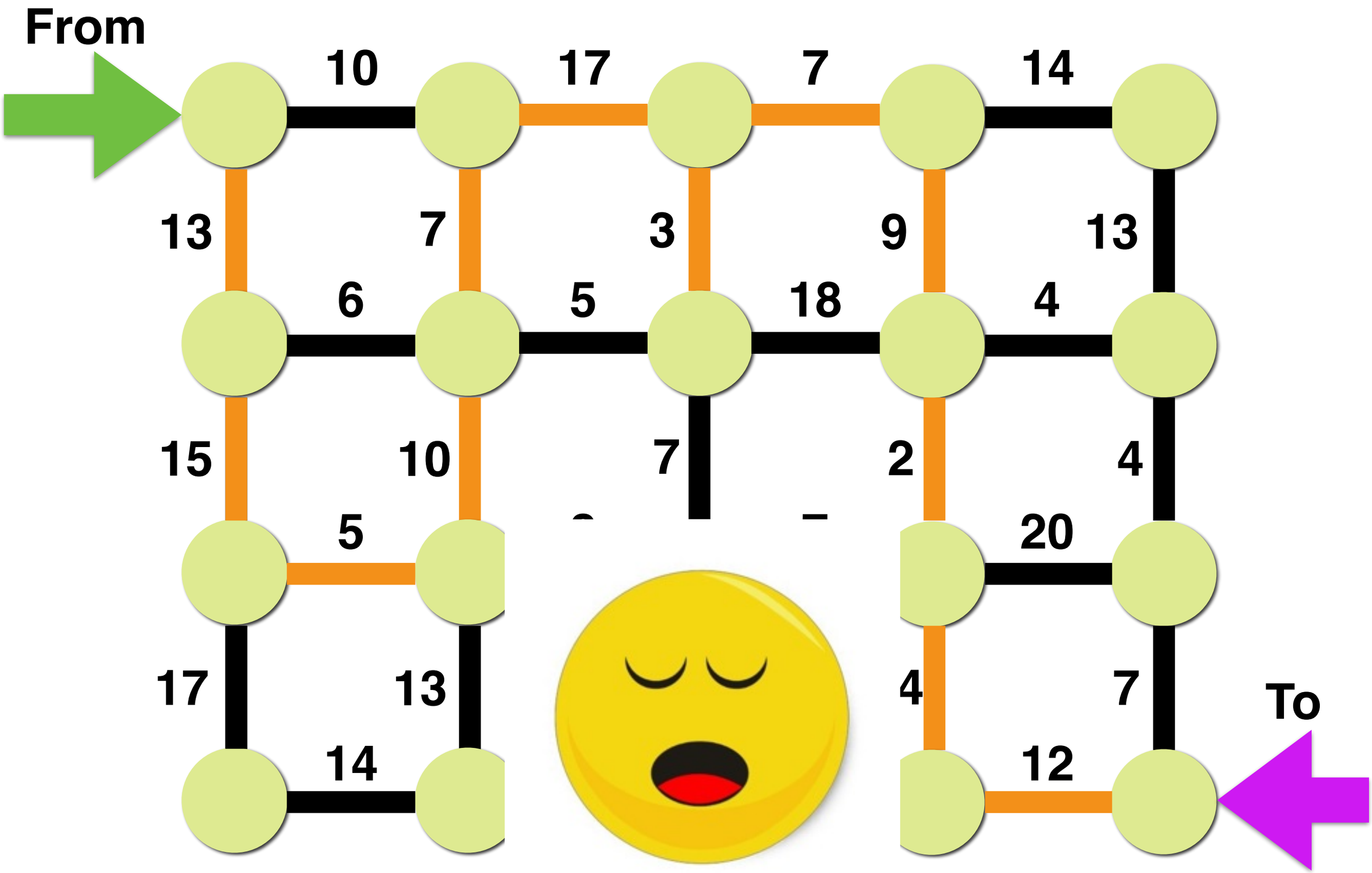


Travel Time: 13 + 6 + 5 + 18 + 2 + 20 + 7 = 71


From
10
17
7
14
13
7
3
9
13
6
5
18
4
15
10
7
2
4
5
8
7
20
17
13
9
14
7
14
11
9
12
To


From
10
17
7
14
13
7
3
9
13
6
5
18
4
15
10
7
2
4
5
8
7
20
17
13
9
14
7
14
11
9
12
To


From
10
17
7
14
13
7
3
9
13
6
5
18
4
15
10
7
2
4
5
8
7
20
17
13
9
14
7
14
11
9
12
To


From
10
17
7
14
13
7
3
9
13
6
5
18
4
15
10
7
2
4
5
20
17
13
14
4
7
12
To

Dijkstar's Algorithm

- It was designed by computer scientist **Edsger W. Dijkstra** in 1956. (May 11, 1930 – August 6, 2002)

- Received the 1972 **A. M. Turing Award**, widely considered the most prestigious award in computer science

Dijkstar's algorithm

Step1: Assign to every node a tentative distance

set zero for the **initial node**

set infinity for **all other nodes**

Step2: Mark all nodes **unvisited**.

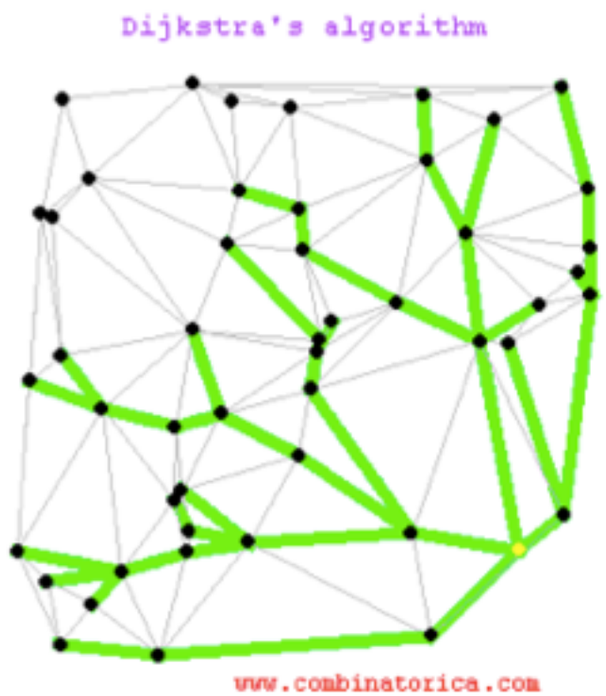
Set the initial node as **current**

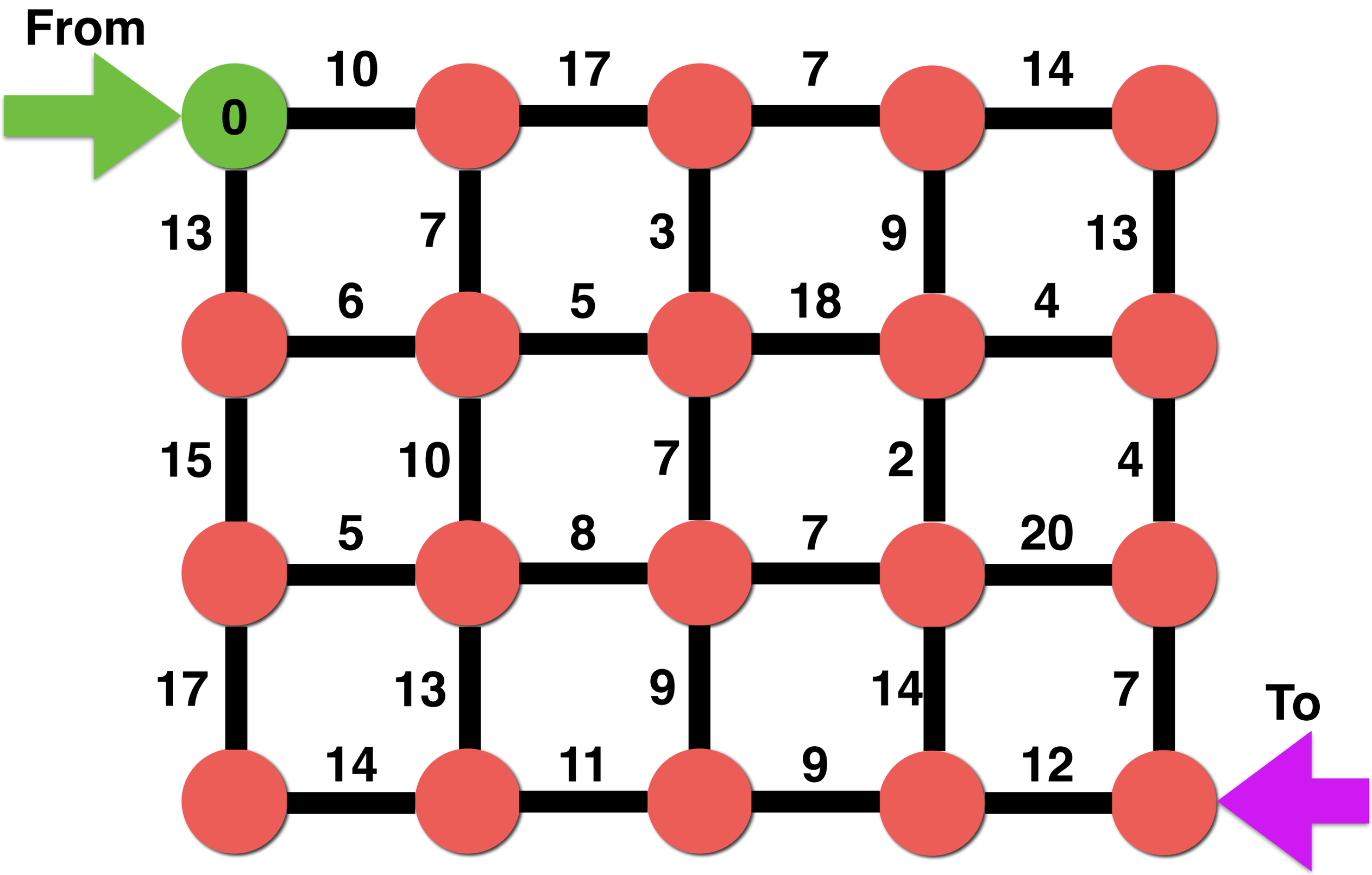
single-source shortest path algorithm

The problem of finding shortest paths from a source vertex v to all other vertices in the graph.

Weighted graph G = (E,V)

Source vertex s ∈ V to all vertices v ∈ V

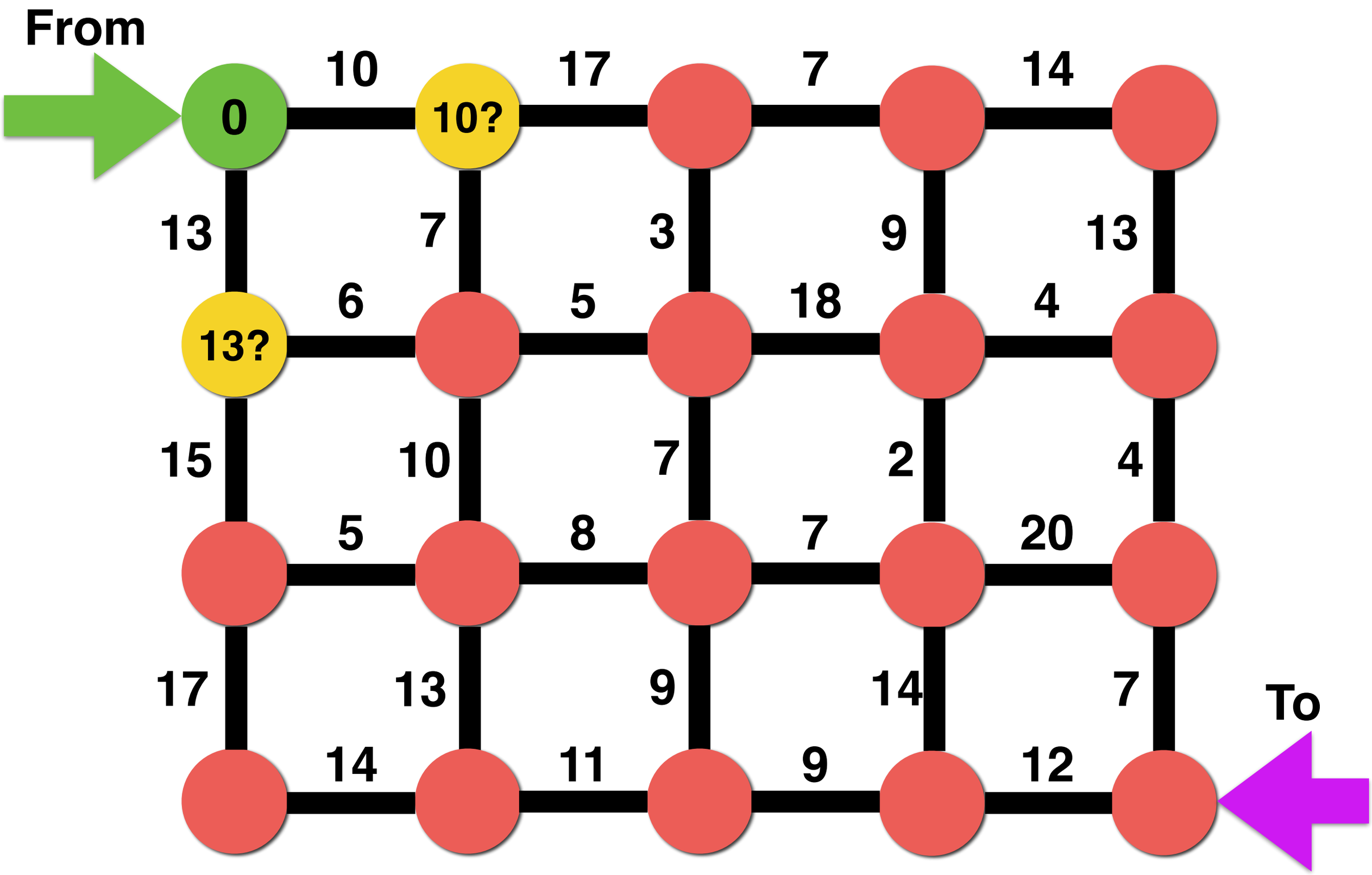



From
0
10
17
7
14
13
7
3
9
13
6
5
18
4
15
10
7
2
4
5
8
7
20
17
13
9
14
7
14
11
9
12
To

Dijkstar's algorithm

Step3: For the current node, **consider all of its unvisited neighbors** and calculate their tentative distances. Compare the newly calculated *tentative* distance to the current assigned value and assign the smaller one.

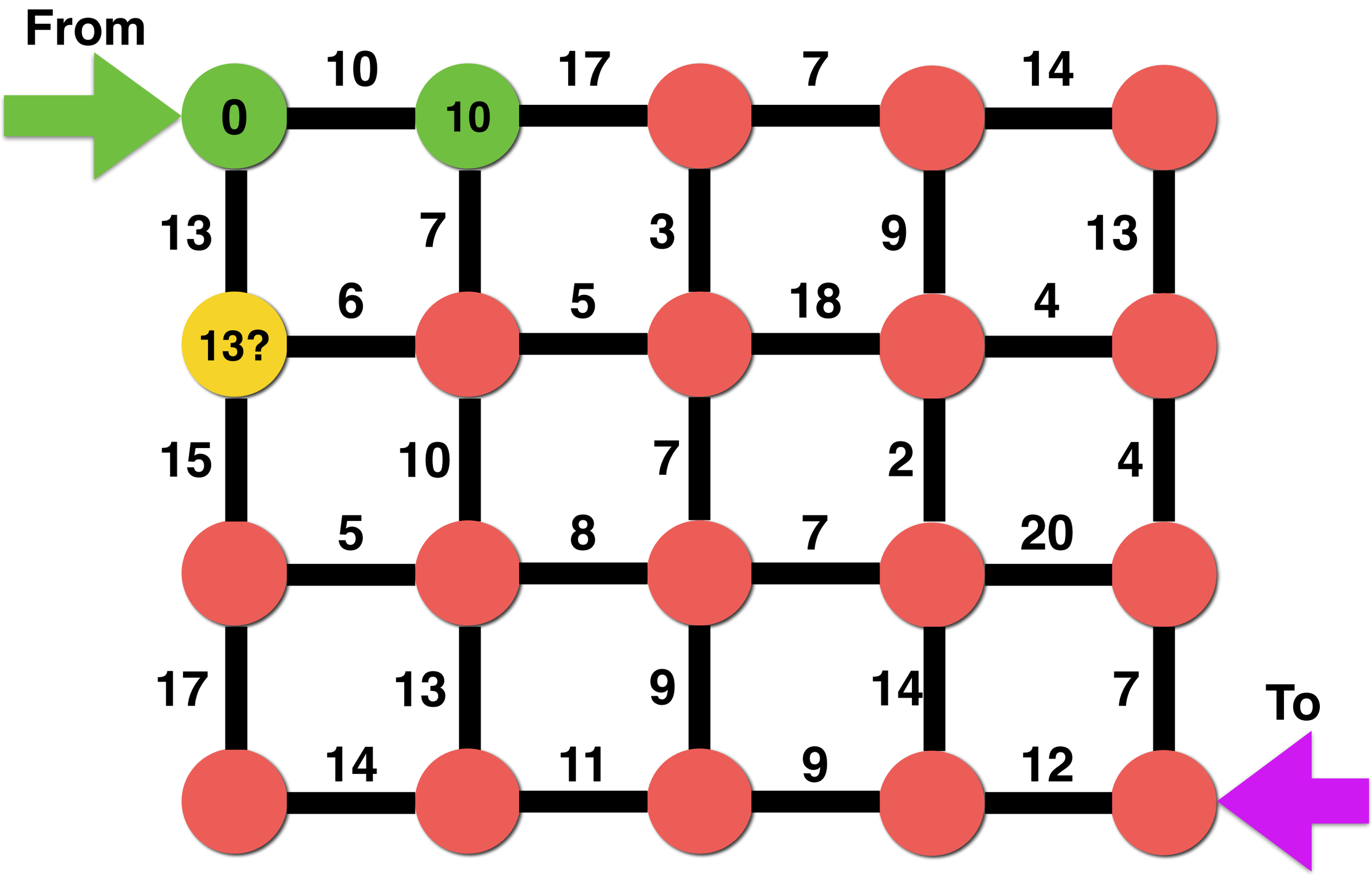
Step4: When we are done considering all of the neighbors of the current node, **mark the current node as visited and remove it from the unvisited set**. A visited node will never be checked again.


From
0
10
10?
17
7
14
13
7
3
9
13
6
13?
5
18
4
15
10
7
2
4
5
8
7
20
17
13
9
14
7
14
11
9
12
To

Dijkstar’s algorithm

Step5: If the destination node has been marked visited the algorithm has finished.

Step6: Select the unvisited node that is marked with the **smallest tentative distance**, and set it as the new “current node”, then go back to step3


From
0
10
17
7
14
10
13
7
3
9
13
13?
6
5
18
4
15
10
7
2
4
5
8
7
20
17
13
9
14
7
14
11
9
12
To

From

0 10 17 7 14

13 7 3 9 13

13? 6 17? 5 18 4

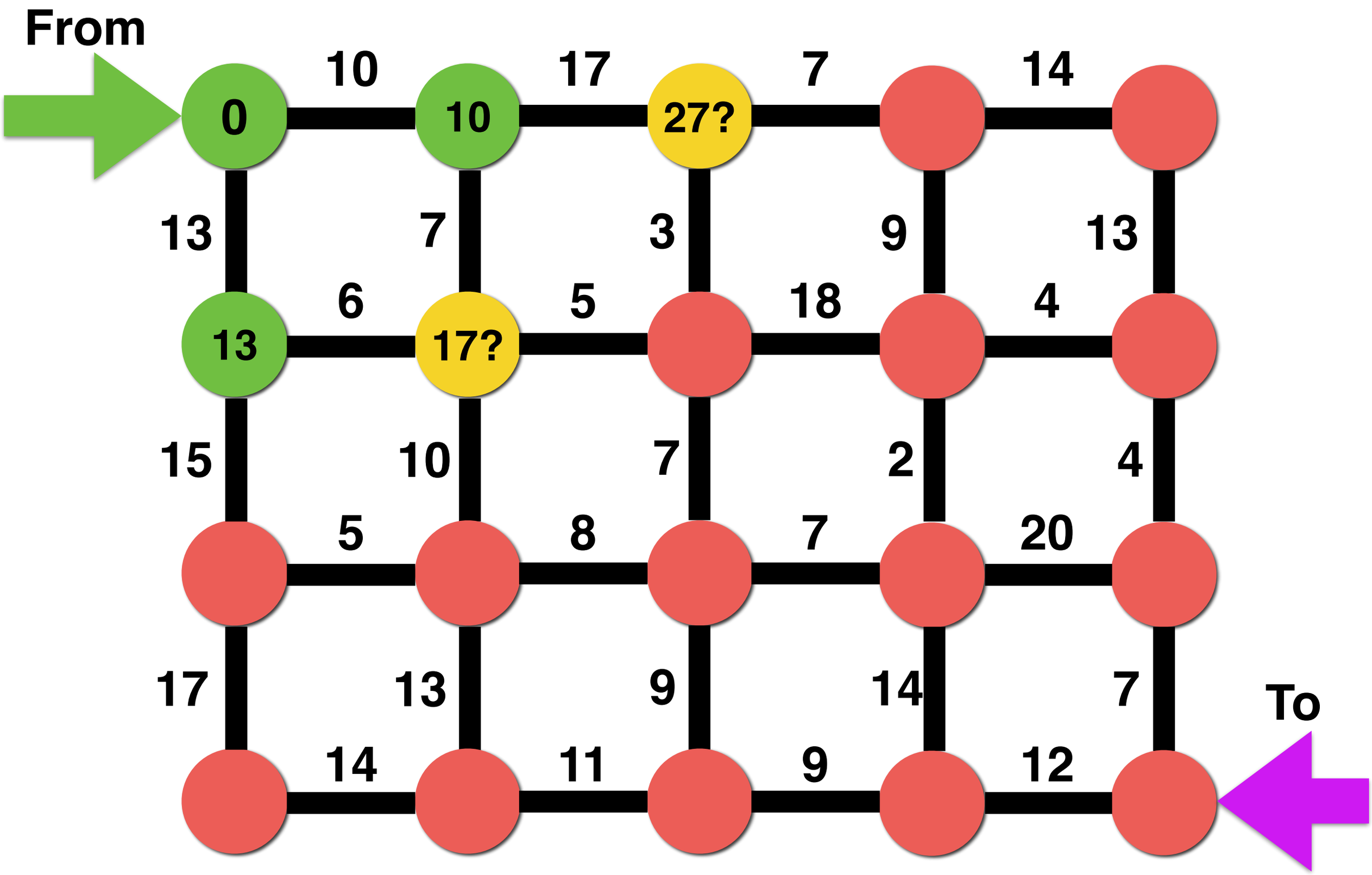
15 10 7 2 4

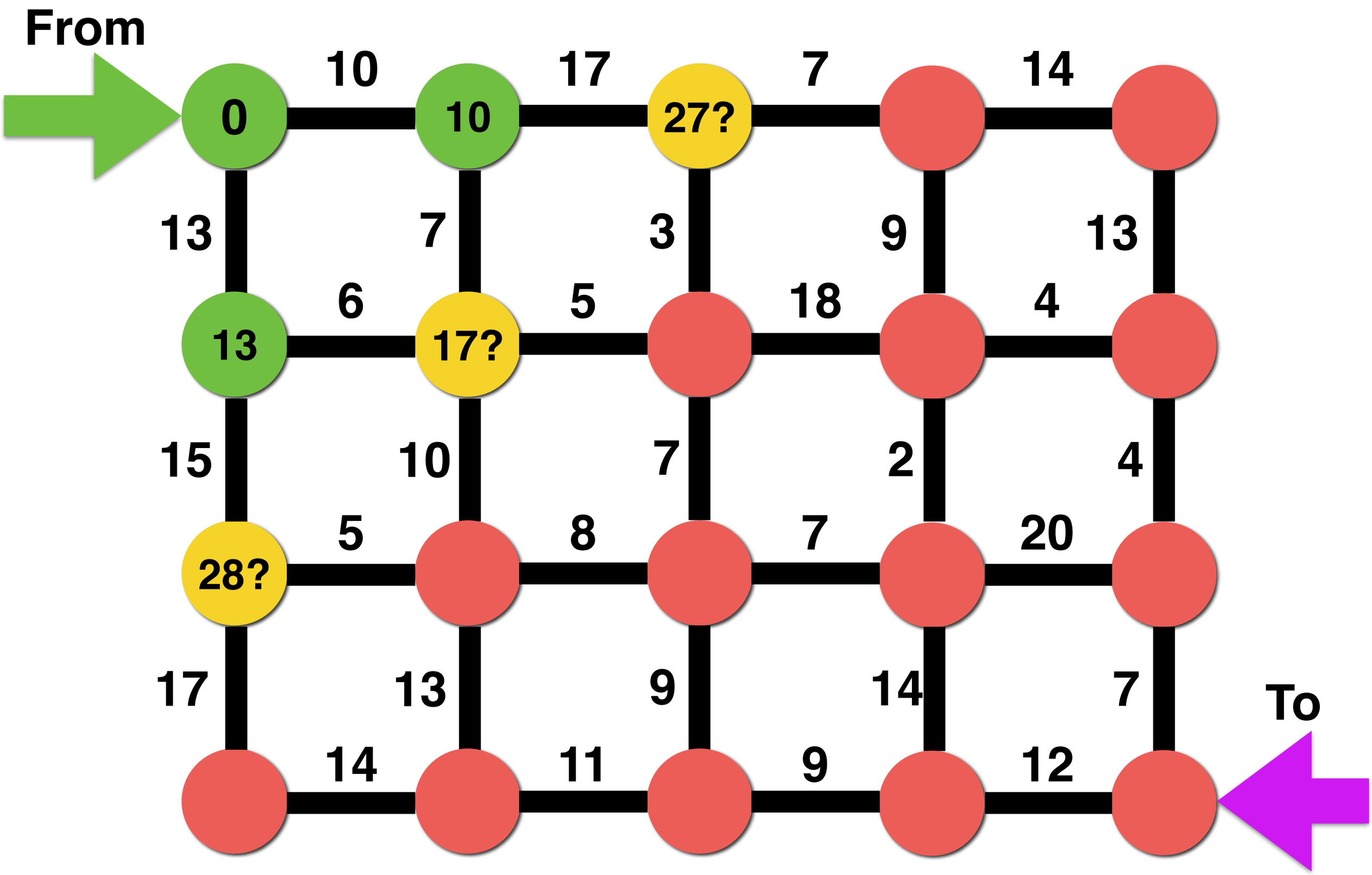
5 8 7 20

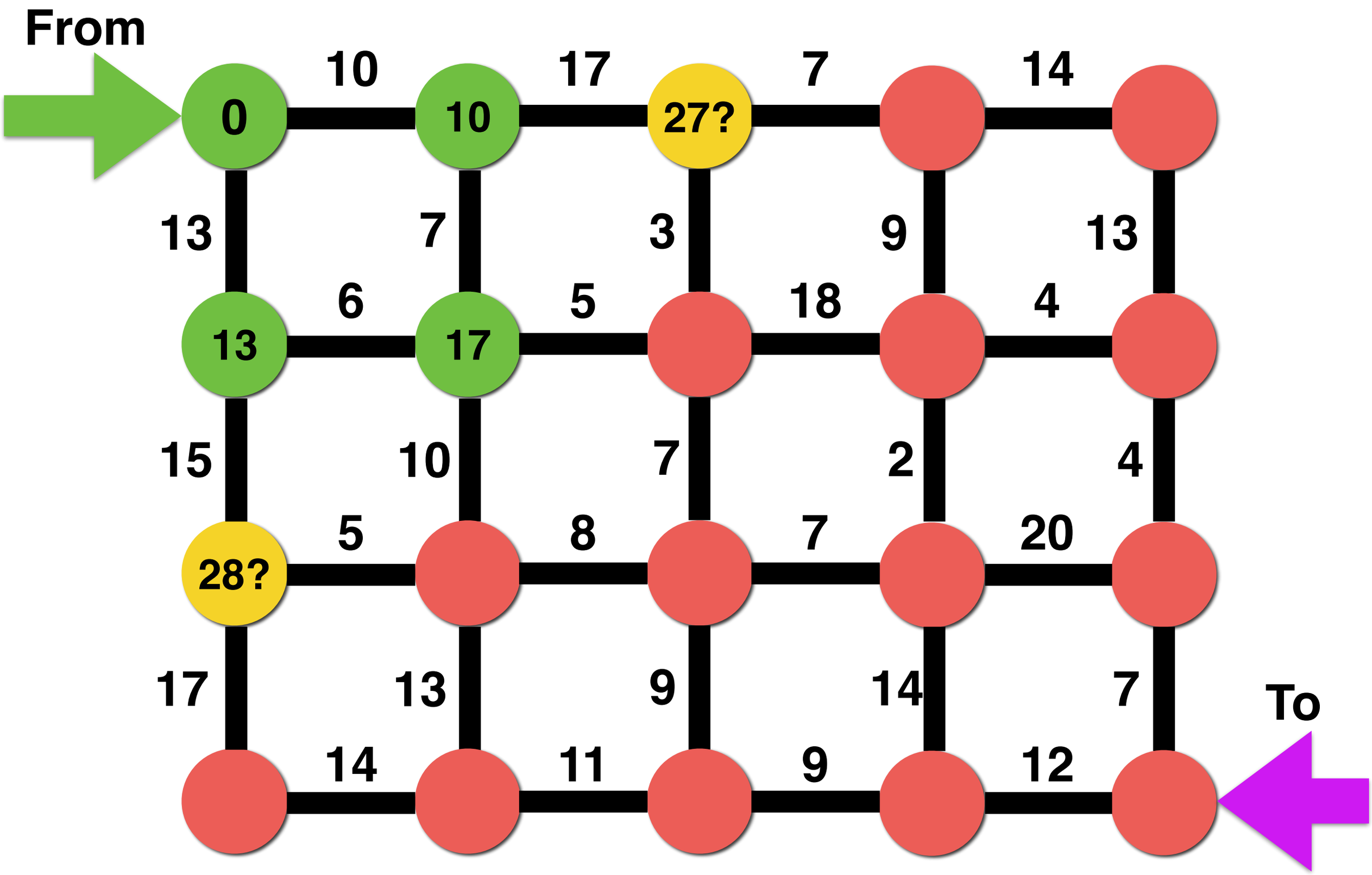
17 13 9 14 7

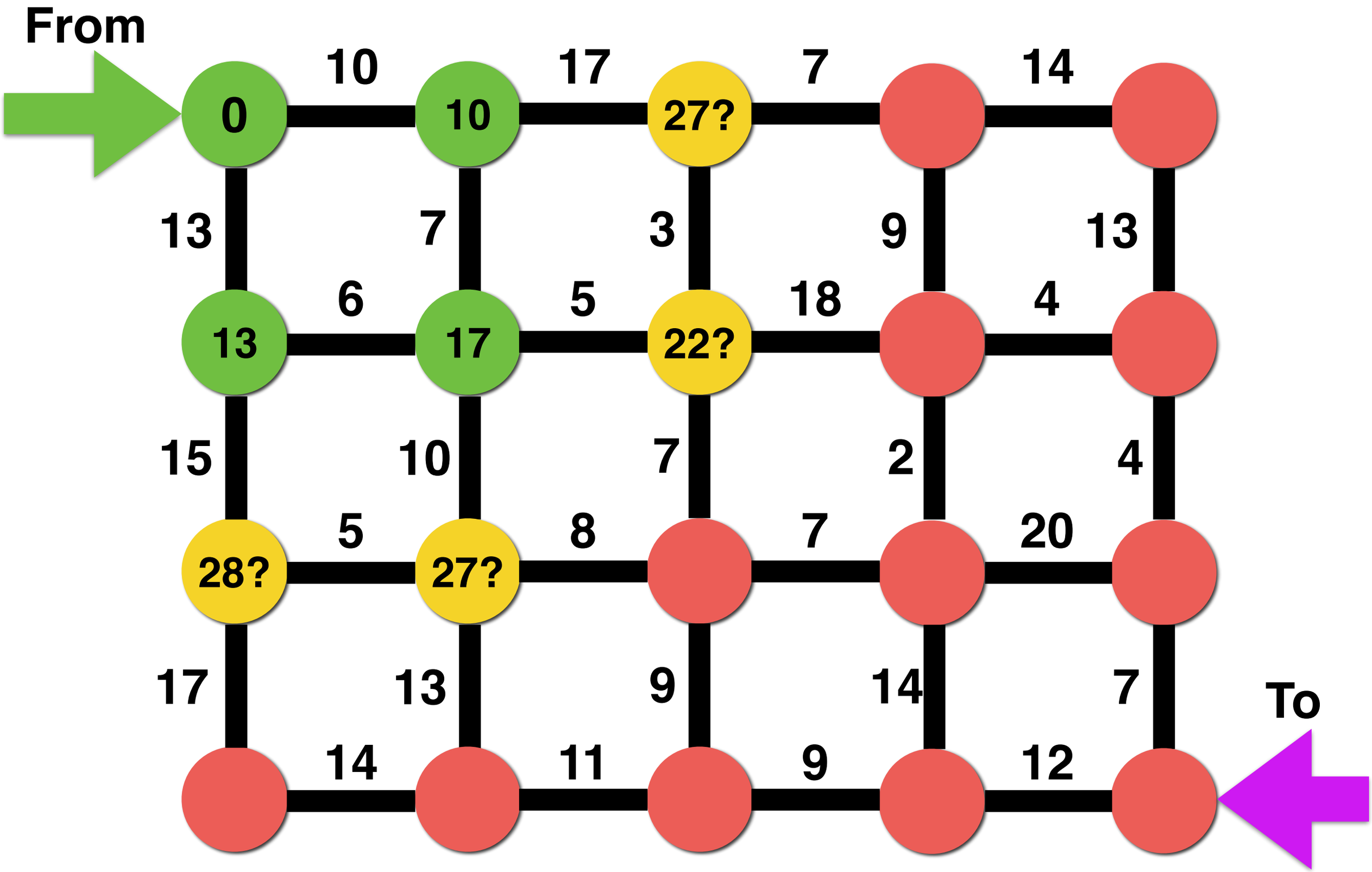
14 11 9 12

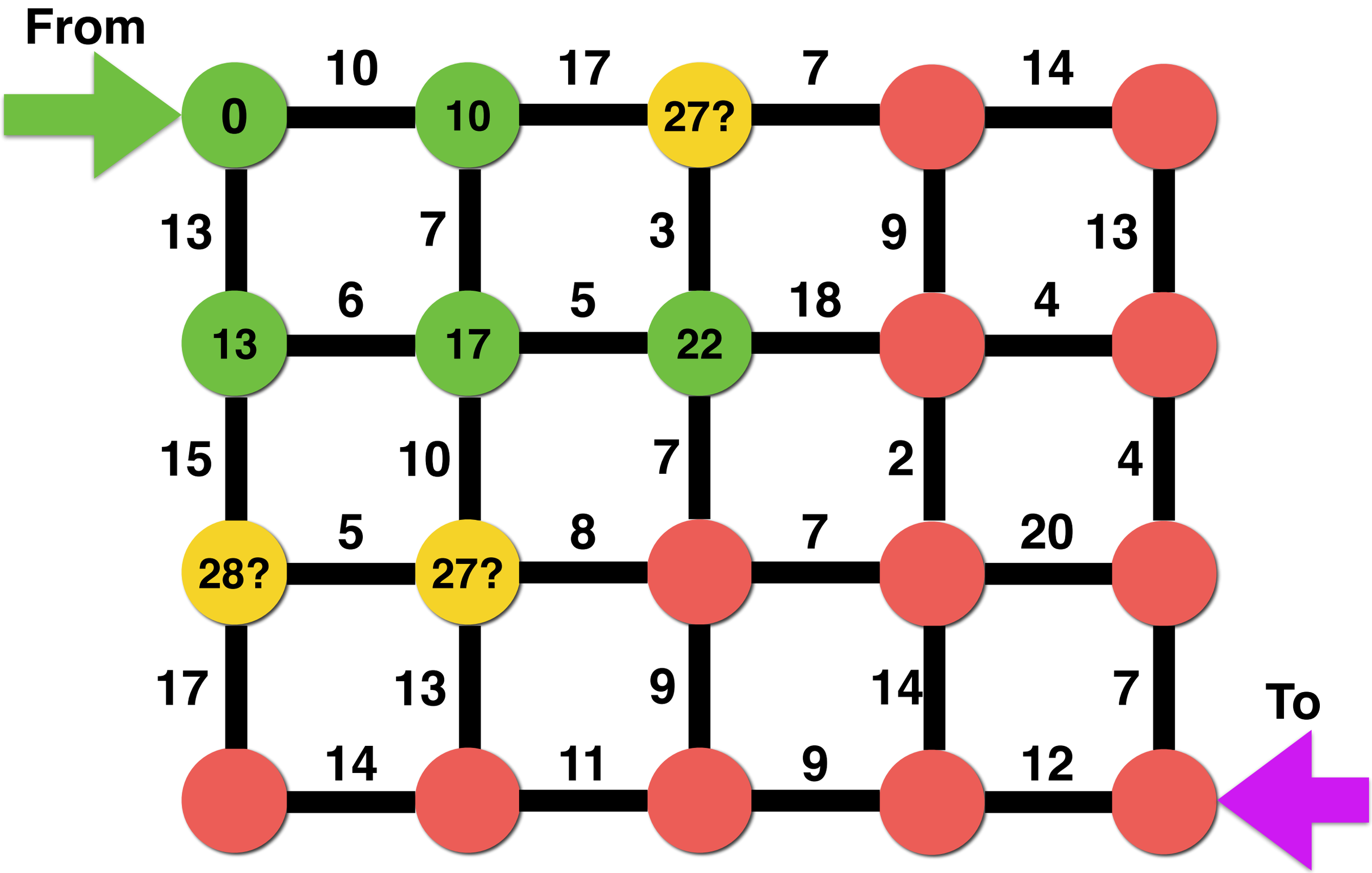
To

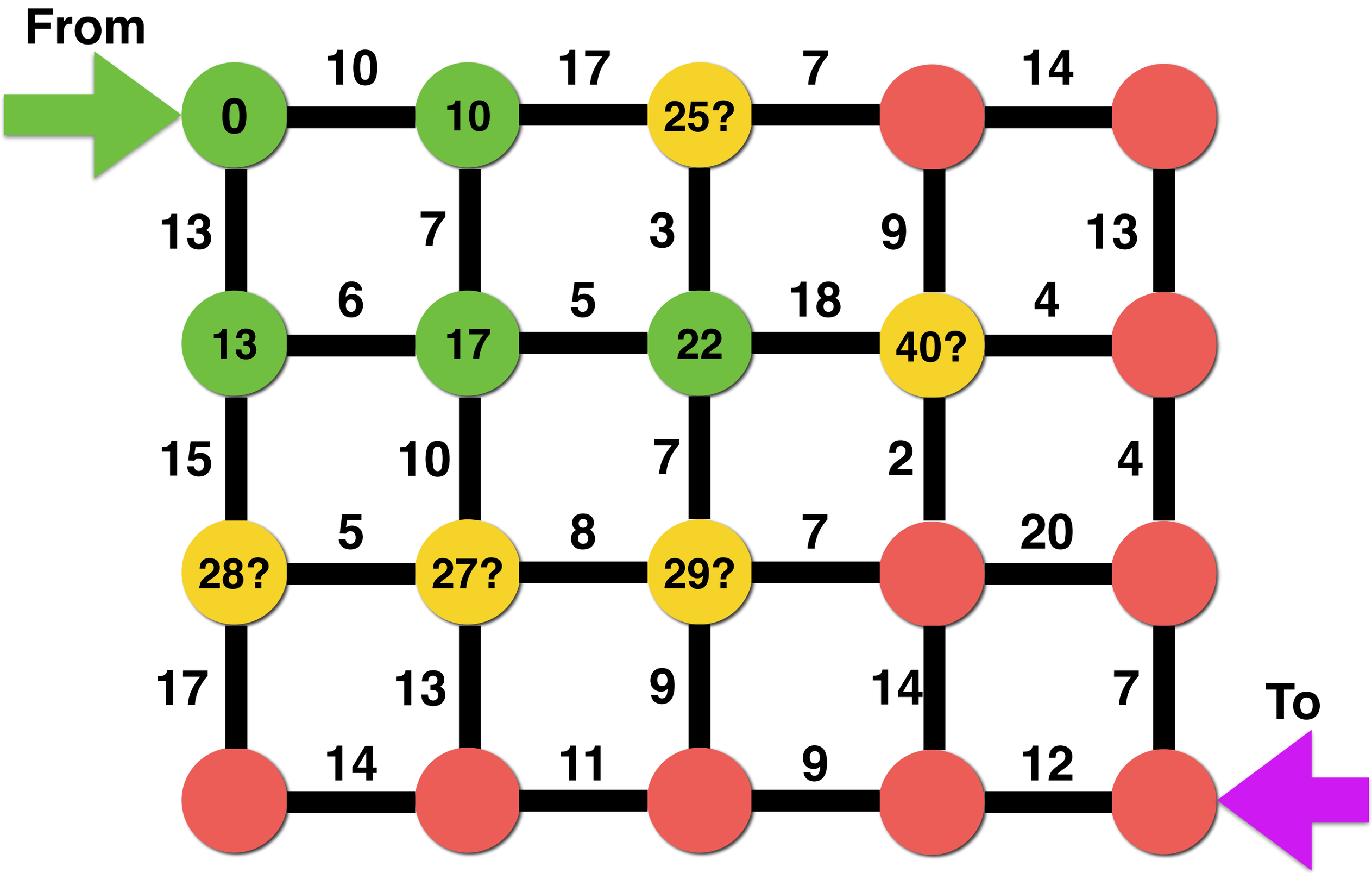

From
0
10
17
27?
7
14
13
7
3
9
13
13
6
17?
5
18
4
15
10
7
2
4
5
8
7
20
17
13
9
14
7
14
11
9
12
To

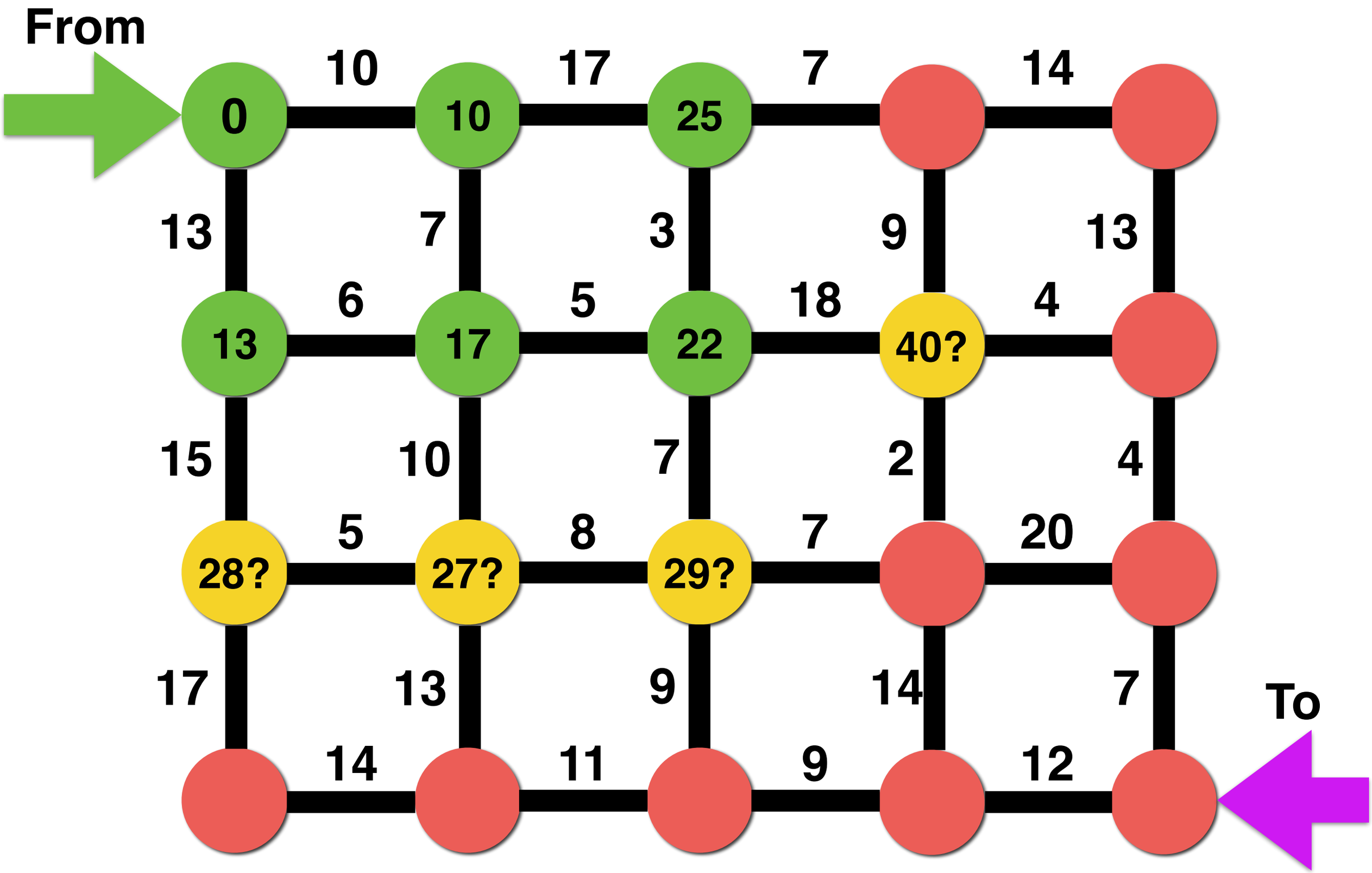

From
0
10
17
7
14
10
27?
13
7
3
9
13
13
6
5
18
4
17?
15
10
7
2
4
5
8
7
20
28?
17
13
9
14
7
To
14
11
9
12

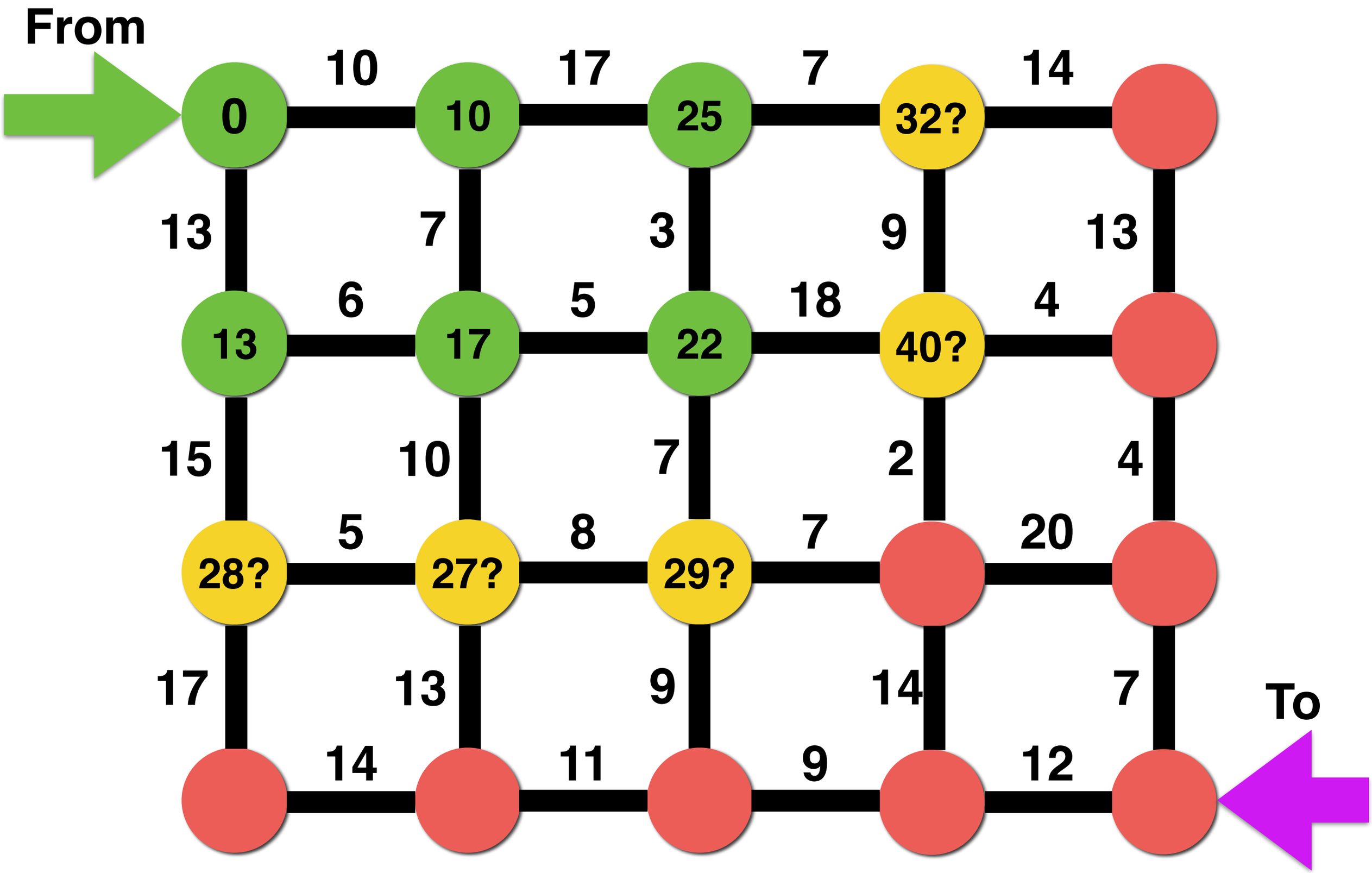

From
0
10
17
27?
7
14
13
7
3
9
13
13
6
17
5
18
4
15
10
7
2
4
28?
5
8
7
20
17
13
9
14
7
14
11
9
12
To

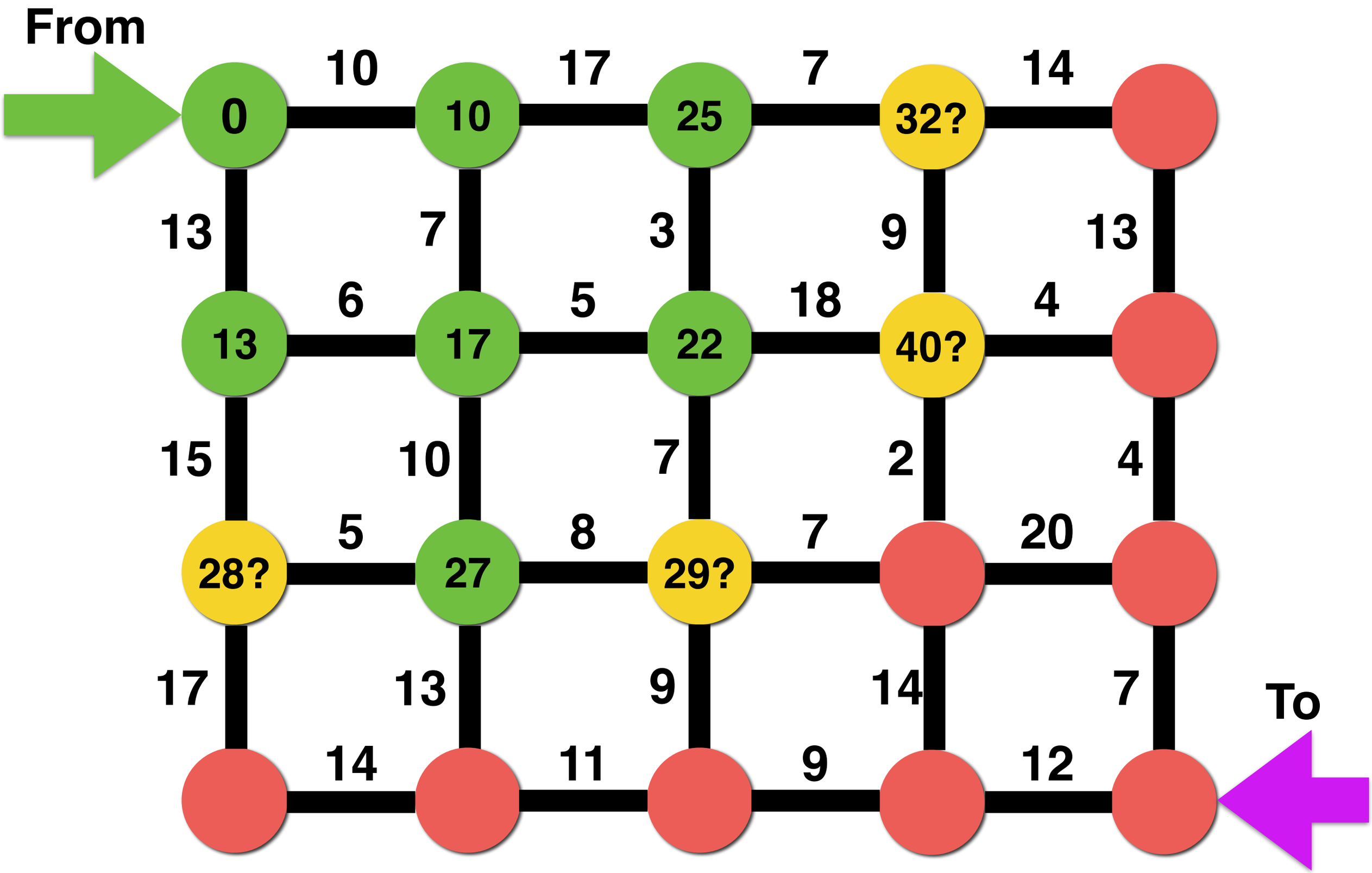

From
0
10
17
27?
7
14
10
13
7
3
9
13
13
6
17
5
22?
18
4
15
10
7
2
4
28?
5
27?
8
7
20
17
13
9
14
7
14
11
9
12
To

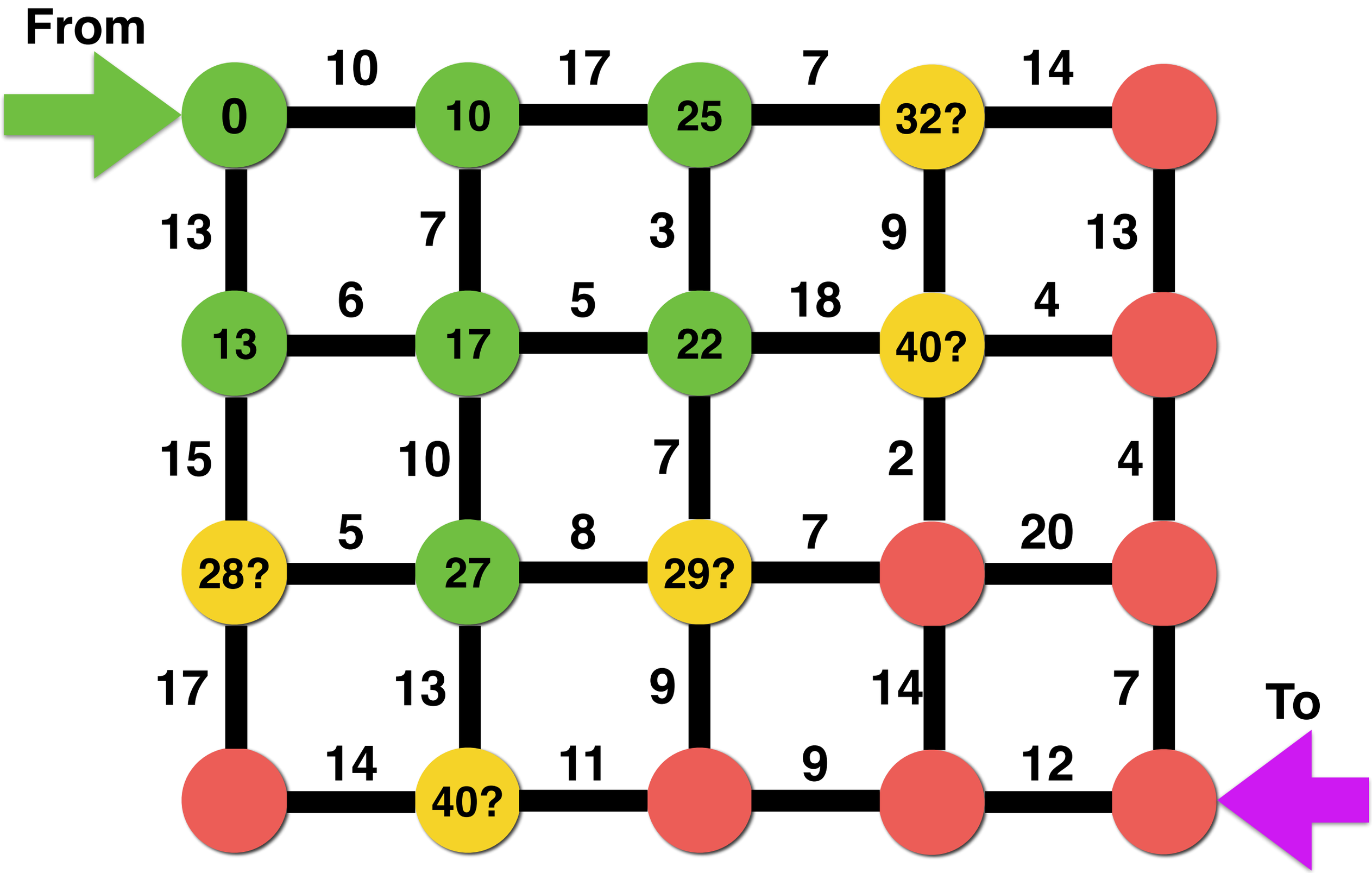

From
0
10
17
27?
7
14
10
13
7
3
9
13
13
6
17
5
22
18
4
15
10
7
2
4
28?
5
27?
8
7
20
17
13
9
14
7
14
11
9
12
To

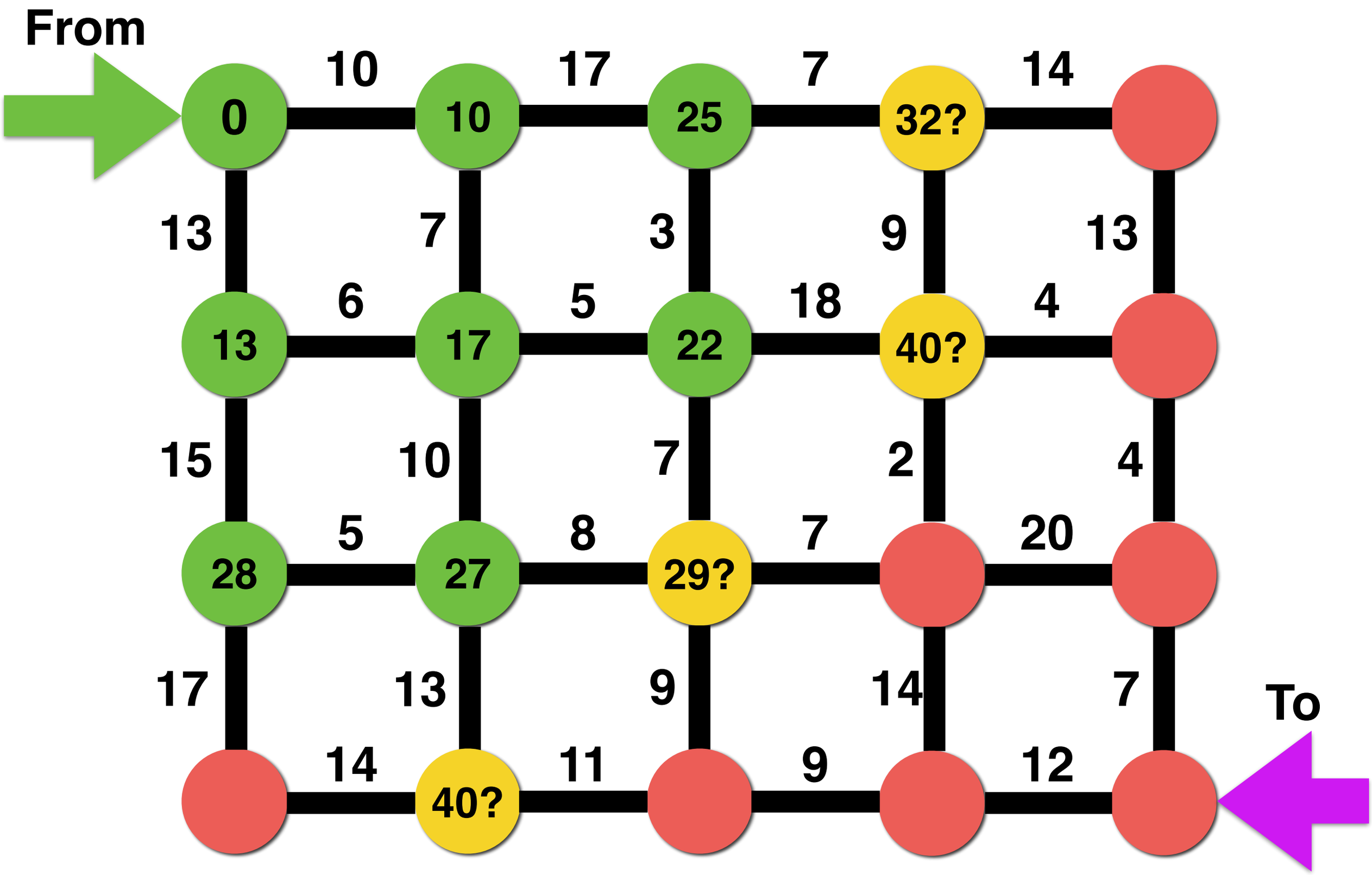

From
0
10
17
25?
7
14
13
7
3
9
13
13
6
17
5
22
18
40?
4
15
10
7
2
4
28?
5
27?
8
29?
7
20
17
13
9
14
7
14
11
9
12
To

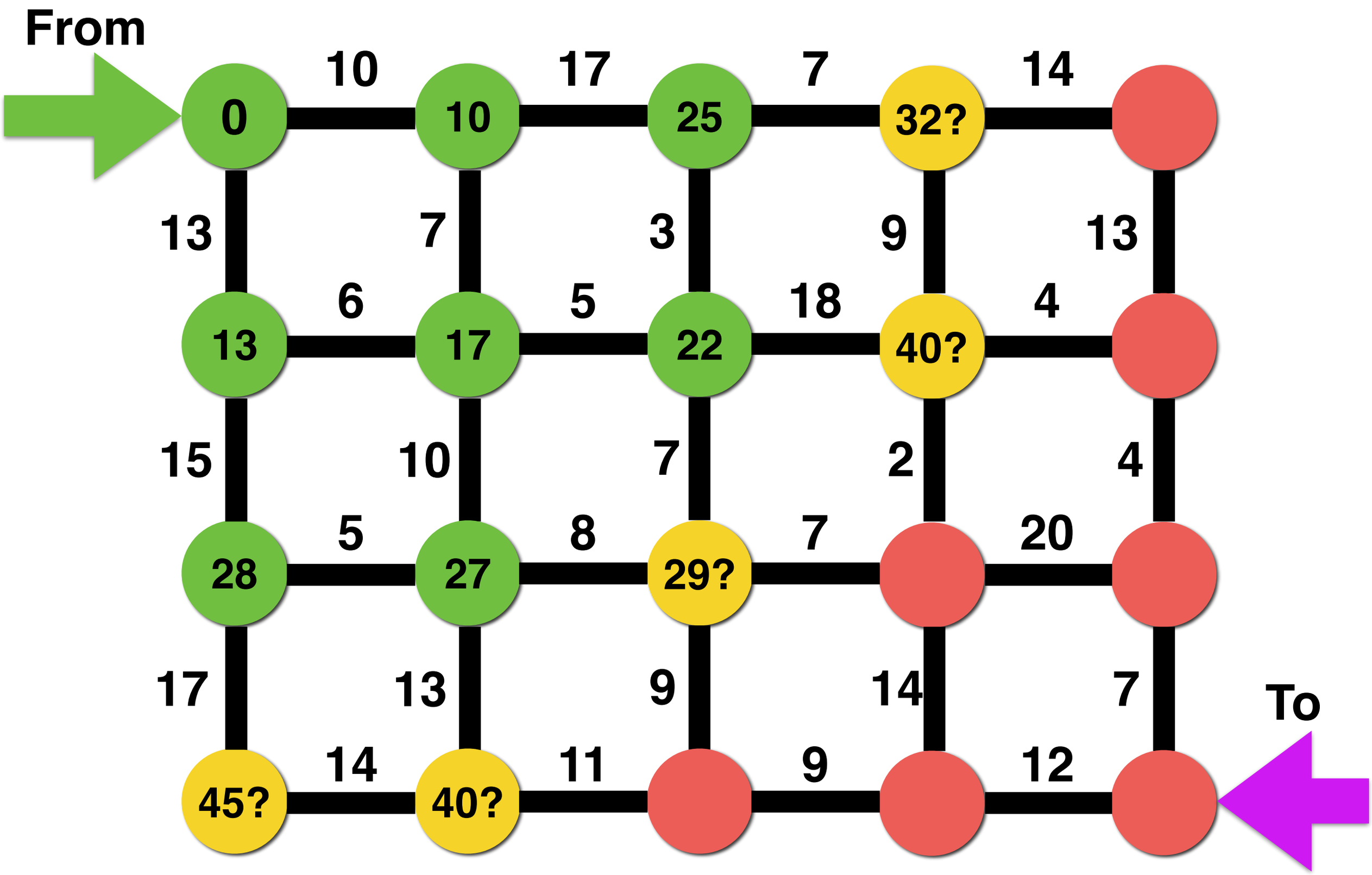

From
0
10
17
7
14
10
25
13
7
3
9
13
6
5
18
4
13
17
22
40?
15
10
7
2
4
5
8
7
20
28?
27?
29?
17
13
9
14
7
To
14
11
9
12

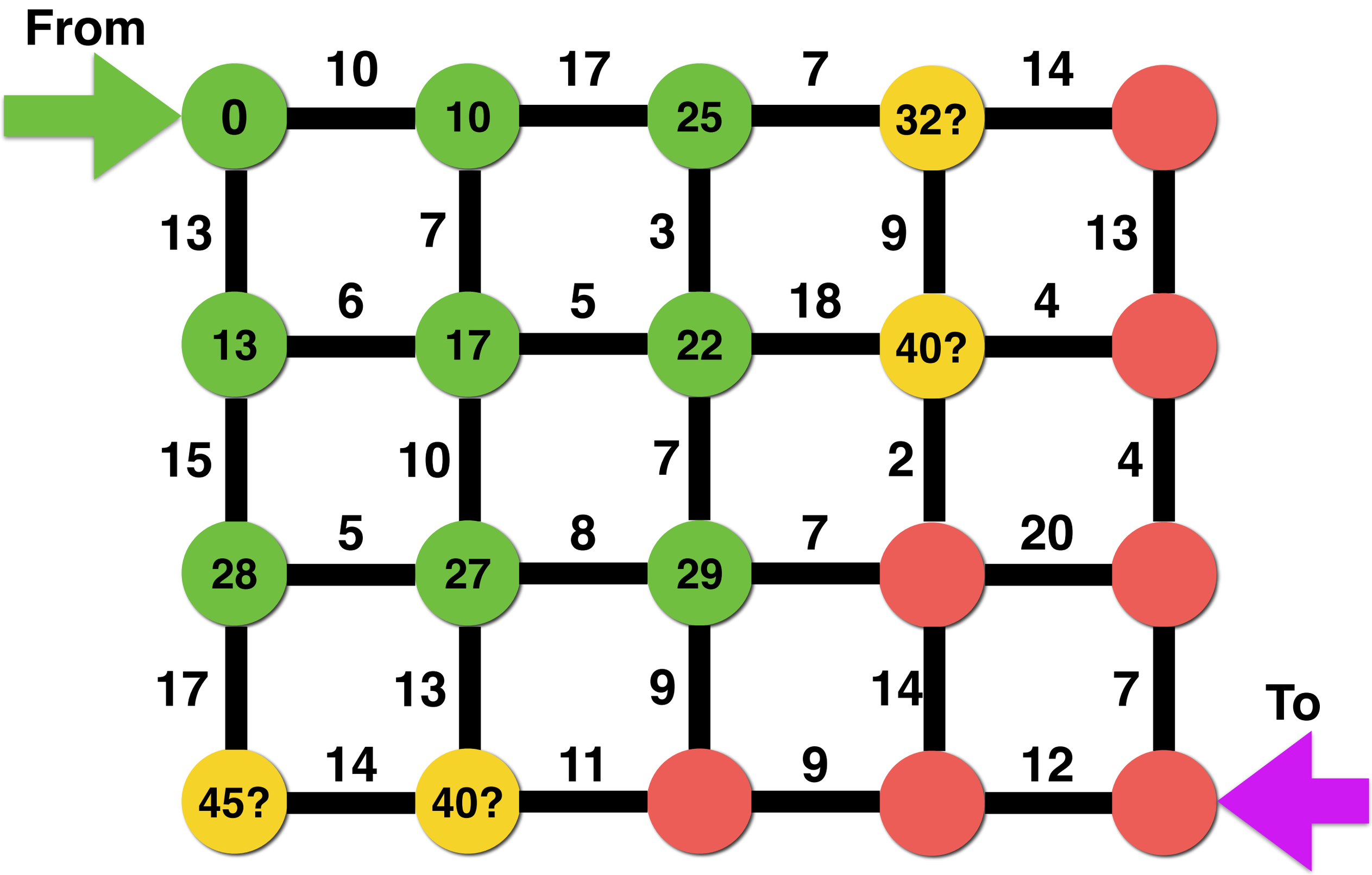

From
0
10
17
7
14
10
25
32?
13
7
3
9
13
6
5
18
4
13
17
22
40?
15
10
7
2
4
5
8
7
20
28?
27?
29?
17
13
9
14
7
14
11
9
12
To


From
10
17
7
14
0
10
25
32?
13
7
3
9
13
6
5
18
4
13
17
22
40?
15
10
7
2
4
5
8
7
20
28?
27
29?
17
13
9
14
7
14
11
9
12
To


From
10
17
7
14
0
10
25
32?
13
7
3
9
13
6
5
18
4
13
17
22
40?
15
10
7
2
4
5
8
7
20
28?
27
29?
17
13
9
14
7
14
11
9
12
40?
To


From
0
10
25
32?
10
17
7
14
13
7
3
9
13
13
6
17
5
22
18
40?
4
15
10
7
2
4
28
5
27
8
29?
7
20
17
13
9
14
7
14
40?
11
9
12
To


From
0
10
25
32?
10
17
7
14
13
7
3
9
13
13
6
17
5
22
18
40?
4
15
10
7
2
4
28
5
27
8
29?
7
20
17
13
9
14
7
45?
14
40?
11
9
12
To


From
0
10
25
32?
10
13
7
3
9
13
13
17
22
40?
6
5
18
4
15
10
7
2
4
28
27
29
5
8
7
20
17
13
9
14
7
45?
40?
14
11
9
12
To

From

0 10 17 7 14

13 7 3 9 13

13 6 17 5 22 18 40? 4

15 10 7 2 4

28 5 27 8 29 7 36? 20

17 13 9 14 7

45? 14 40? 11 38? 9 12

To

From

0 10 10 17 25 7 32 14

13 7 3 9 13

13 6 17 5 22 18 40? 4

15 10 7 2 4

28 5 27 8 29 7 36? 20

17 13 9 14 7

45? 14 40? 11 38? 9 12

To

From

0 10 10 17 25 7 32 14 46?

13 7 3 9 13

13 6 17 5 22 18 40? 4

15 10 7 2 4

28 5 27 8 29 7 36? 20

17 13 9 14 7

45? 14 40? 11 38? 9 12

To

From

0 10 17 7 14 46?

13 7 3 9 13

13 6 17 5 22 18 40? 4

15 10 7 2 4

28 5 27 8 29 7 36 20

17 13 9 14 7

45? 14 40? 11 38? 9 12

To

From

0 10 10 17 25 7 32 14 46?

13 7 3 9 13

13 6 17 5 22 18 38? 4

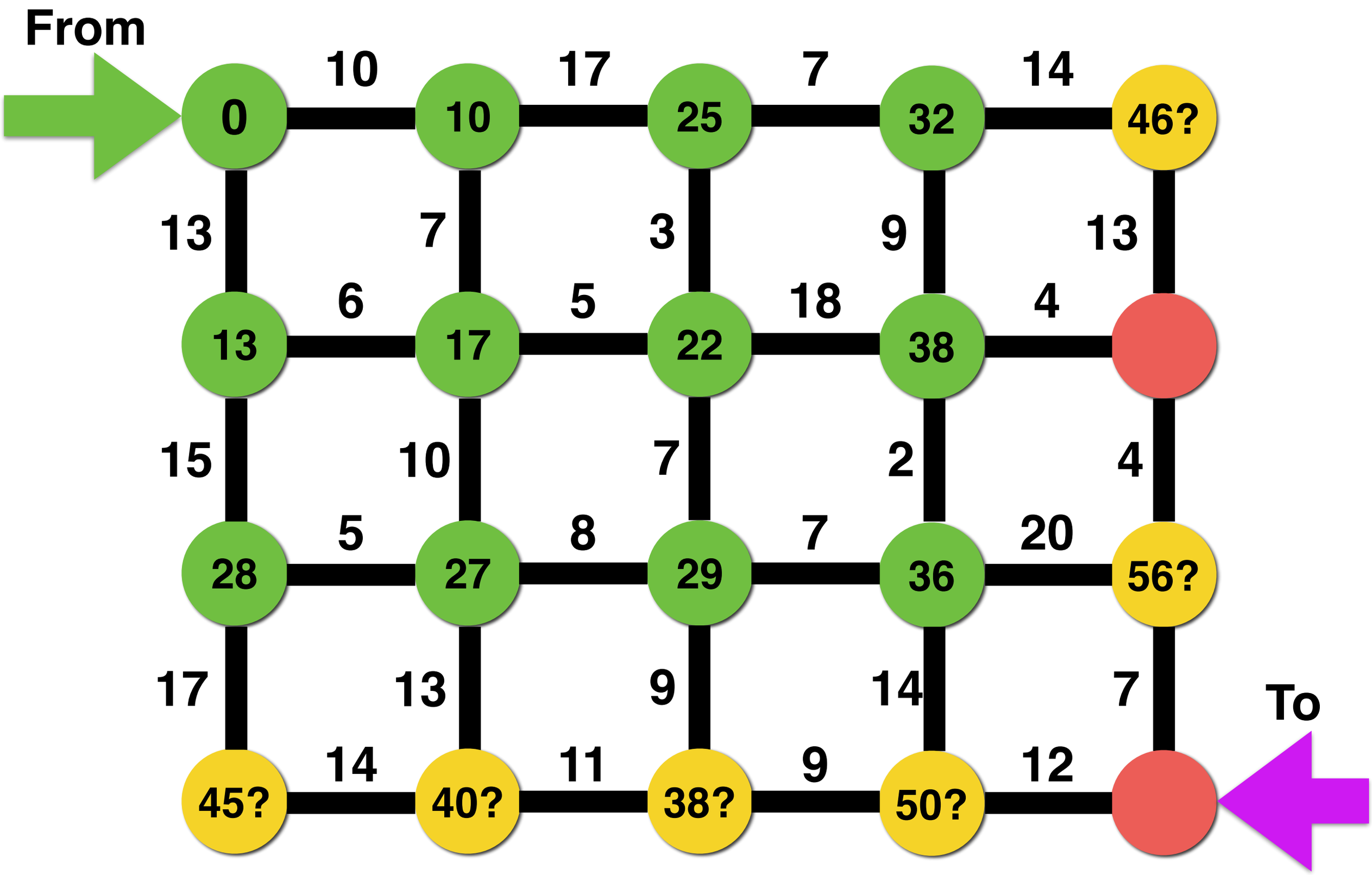
15 10 7 2 4

28 5 27 8 29 7 36 20 56?

17 13 9 14 7

45? 14 40? 11 38? 9 50? 12

To


From
0
10
25
32
46?
13
13
6
17
5
22
18
38
4
15
10
7
2
4
28
5
27
8
29
7
36
20
56?
17
13
9
14
7
45?
14
40?
11
38?
9
50?
12
To

From

0 10 10 17 25 7 32 14 46?

13 7 3 9 13

13 6 17 5 22 18 38 4 42?

15 10 7 2 4

28 5 27 8 29 7 36 20 56?

17 13 9 14 7

45? 14 40? 11 38? 9 50? 12

To

From

0 10 17 7 14 46?

13 7 3 9 13

13 6 17 5 22 18 38 4 42?

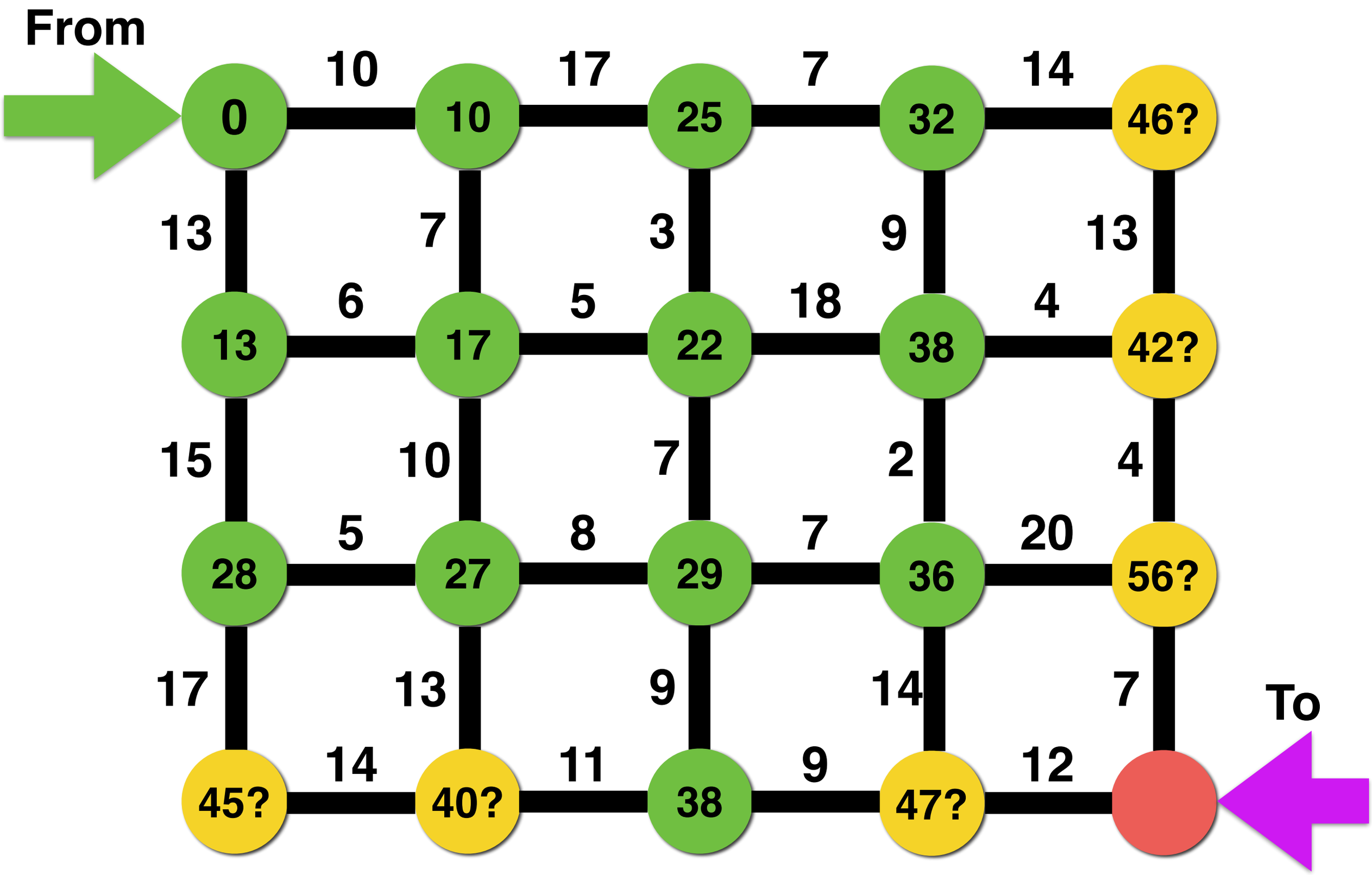
15 10 7 2 4

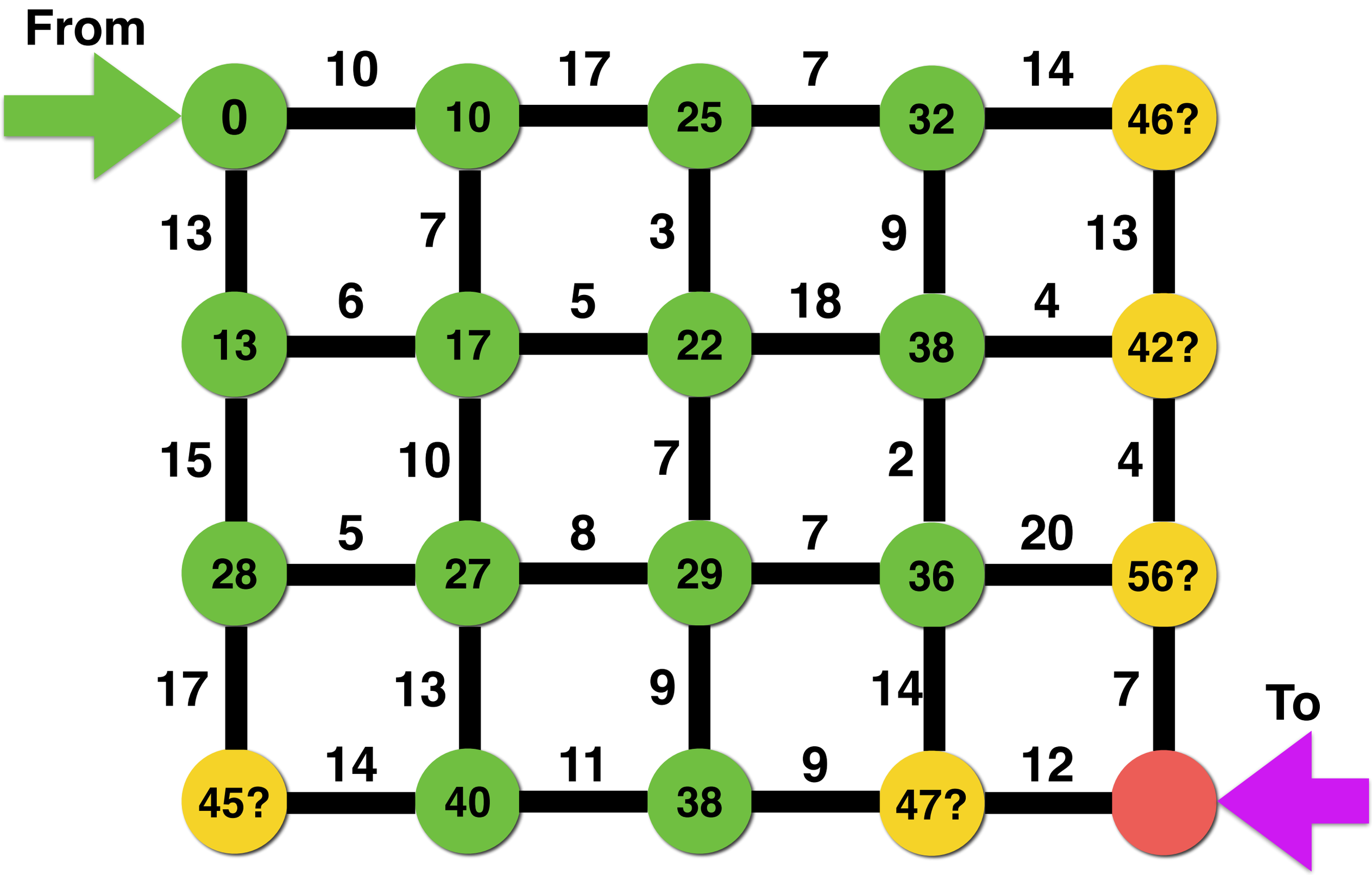
28 5 27 8 29 7 36 20 56?

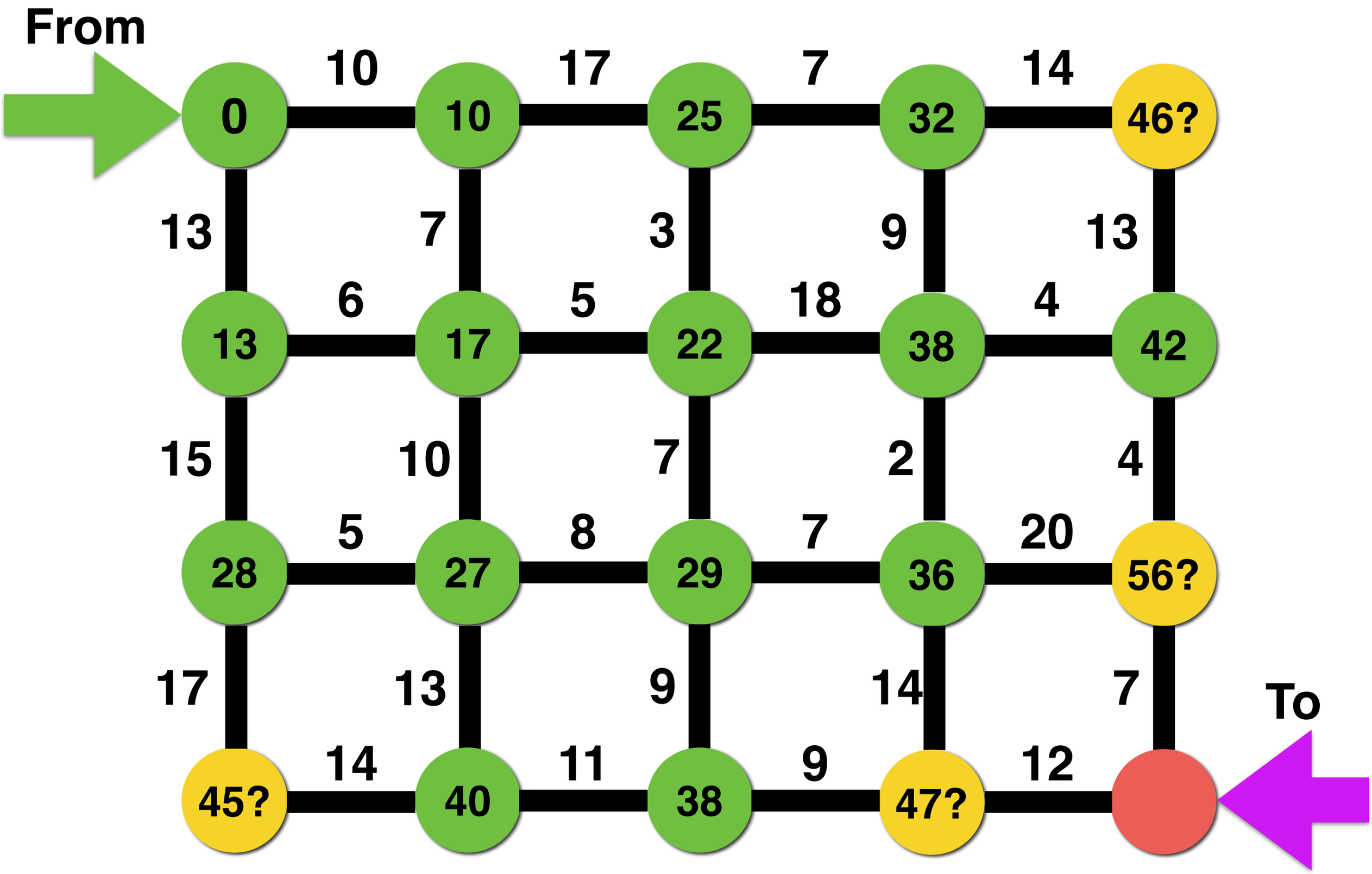
17 13 9 14 7

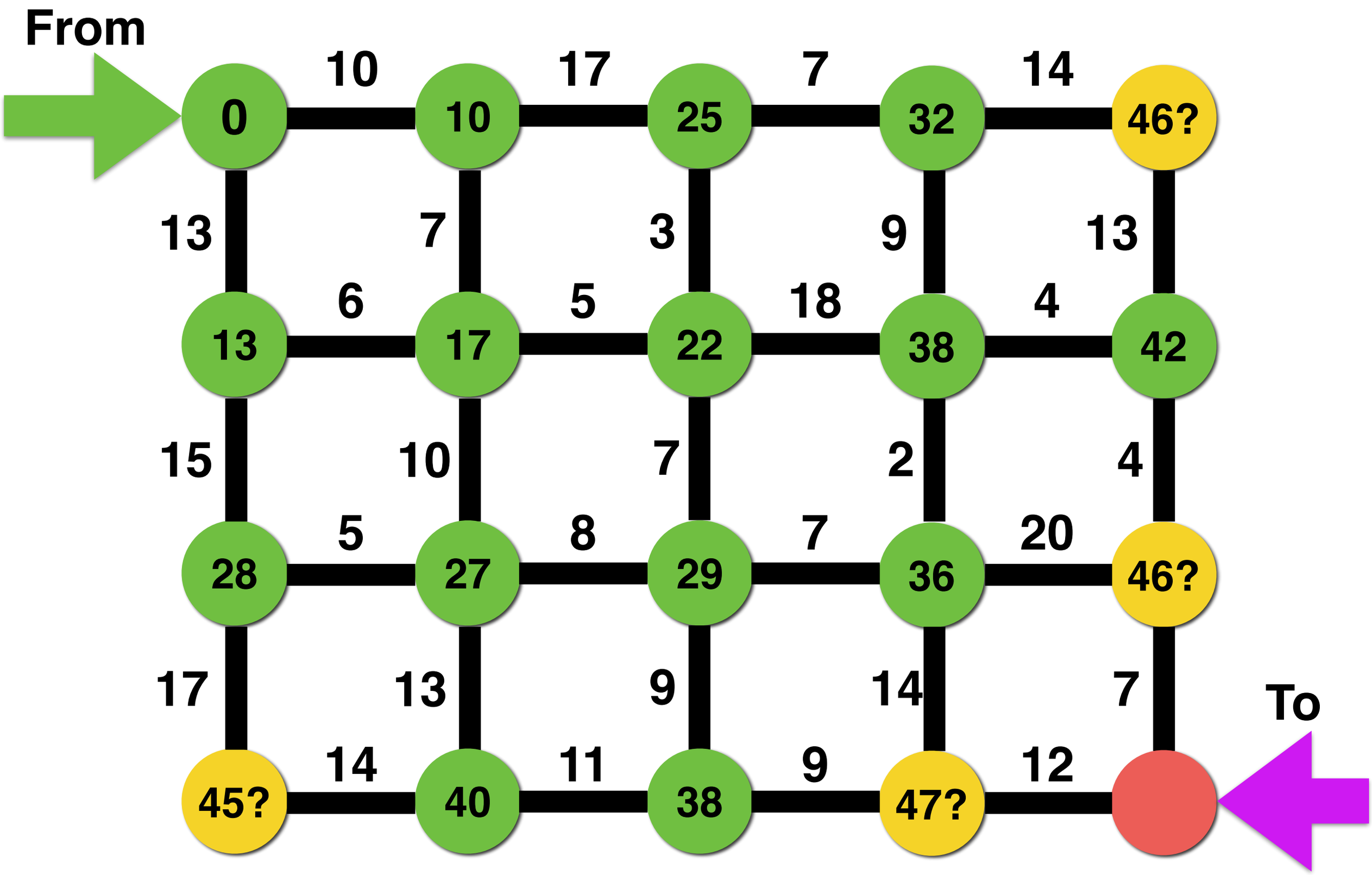
45? 14 40? 11 38 9 50? 12

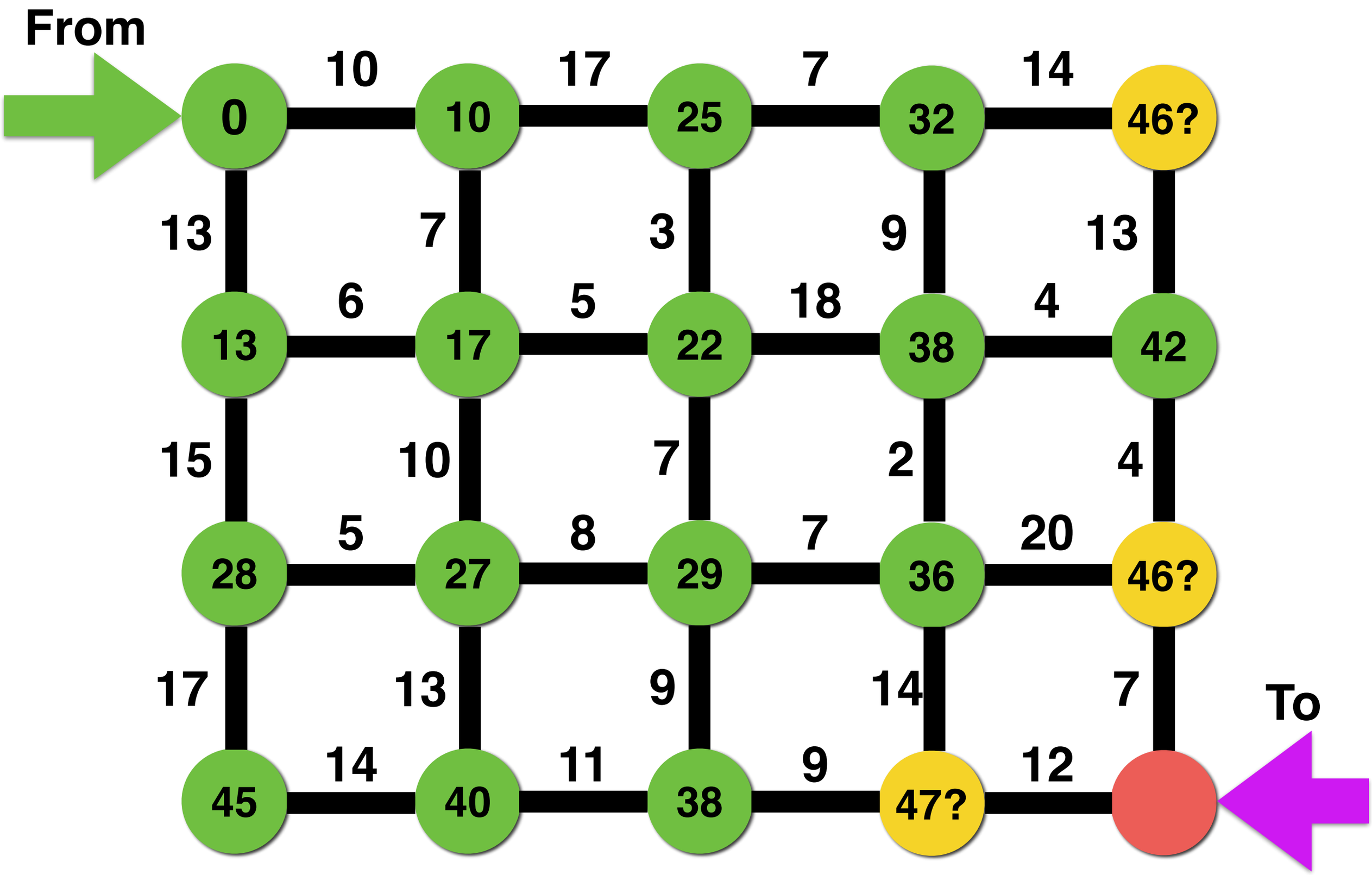
To

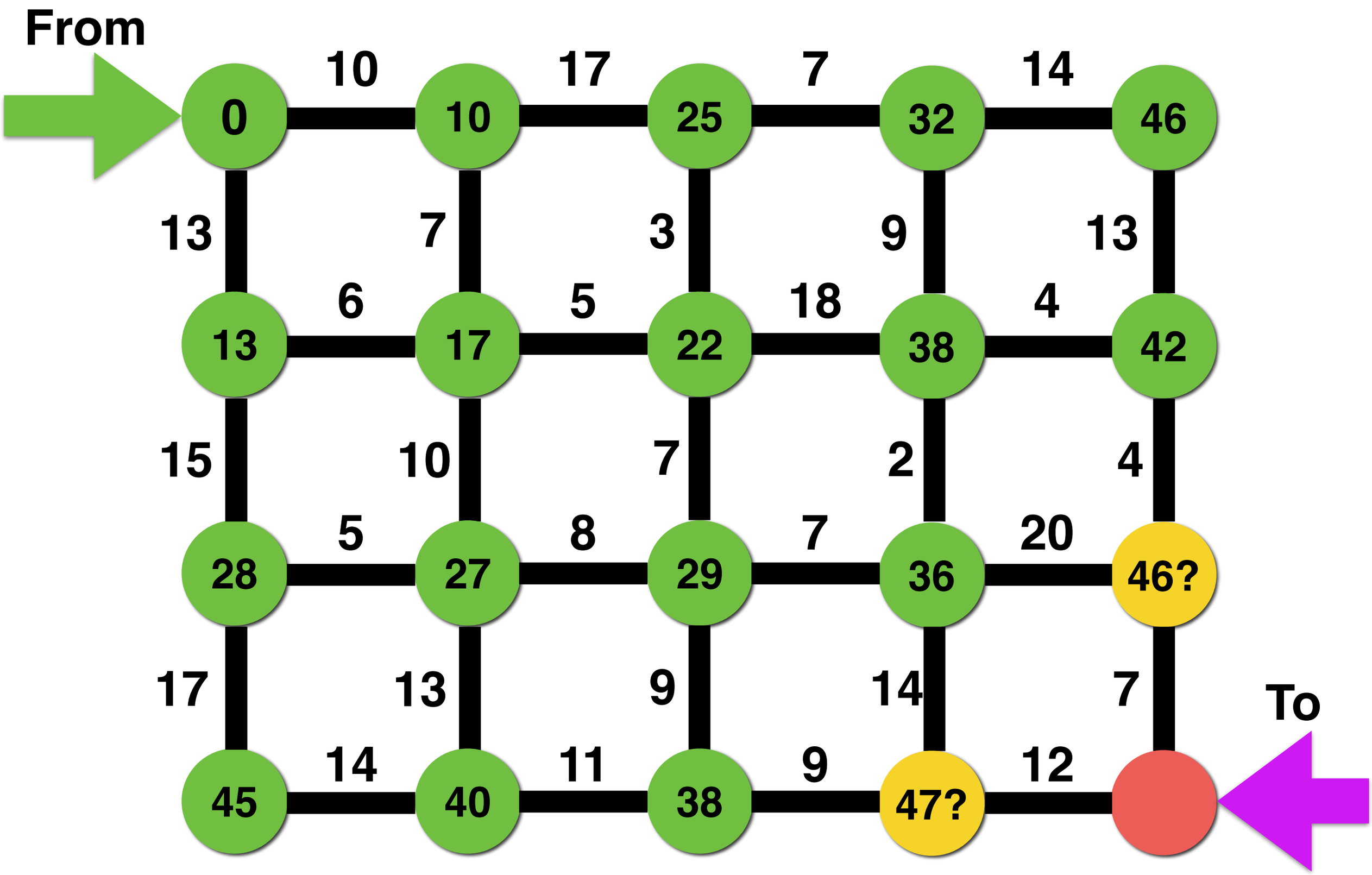

From
0
10
25
32
46?
13
13
6
17
5
22
18
38
4
42?
15
10
7
2
4
28
5
27
8
29
7
36
20
56?
17
13
9
14
7
45?
14
40?
11
38
9
47?
12
To

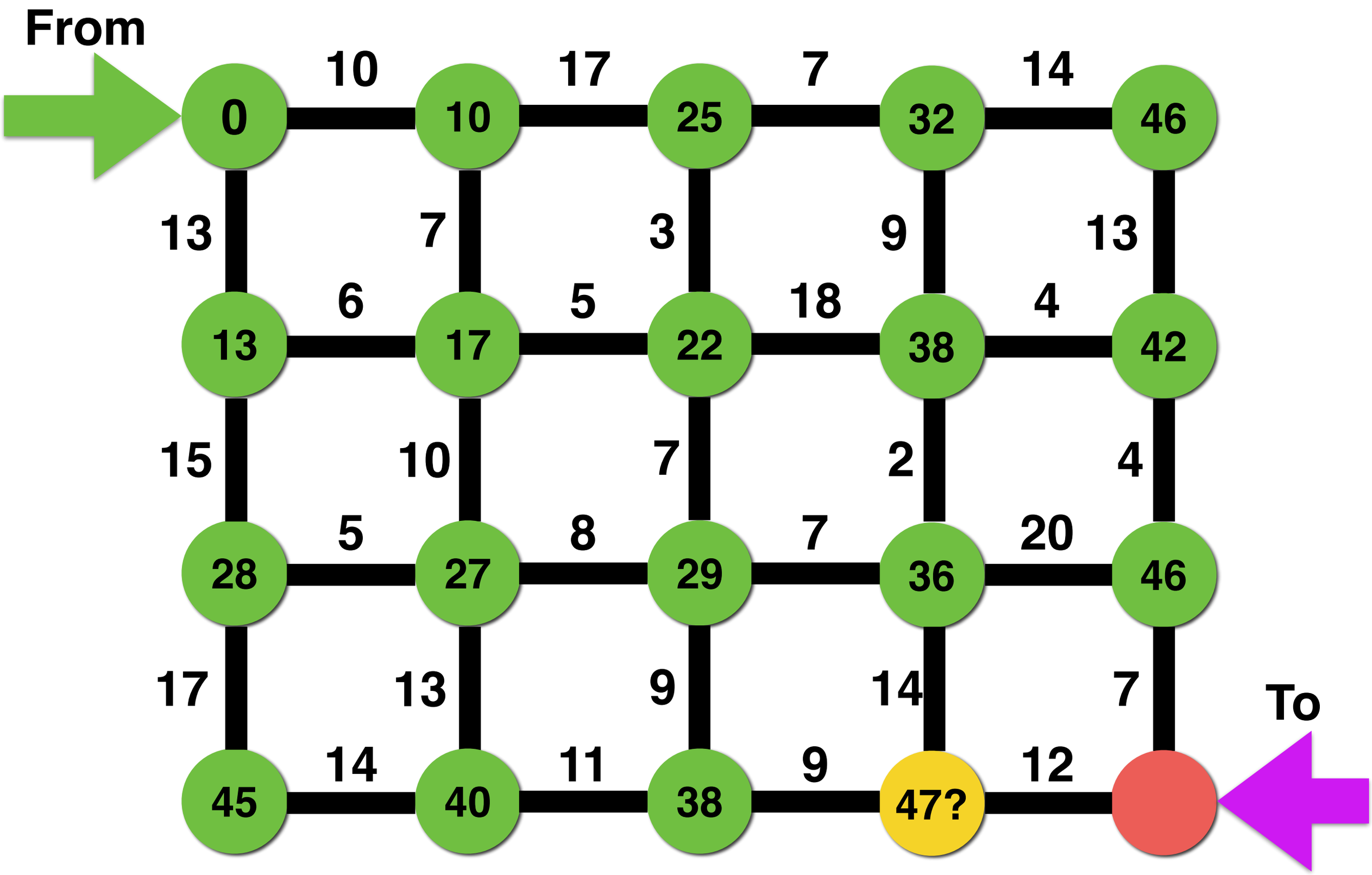

From
0
10
25
32
46?
13
13
6
18
4
42?
38
22
17
10
28
27
29
36
56?
17
14
45?
40
38
47?
To
10
17
7
14
13
7
3
9
13
6
5
18
4
15
10
7
2
4
5
8
7
20
13
9
14
7
14
11
9
12

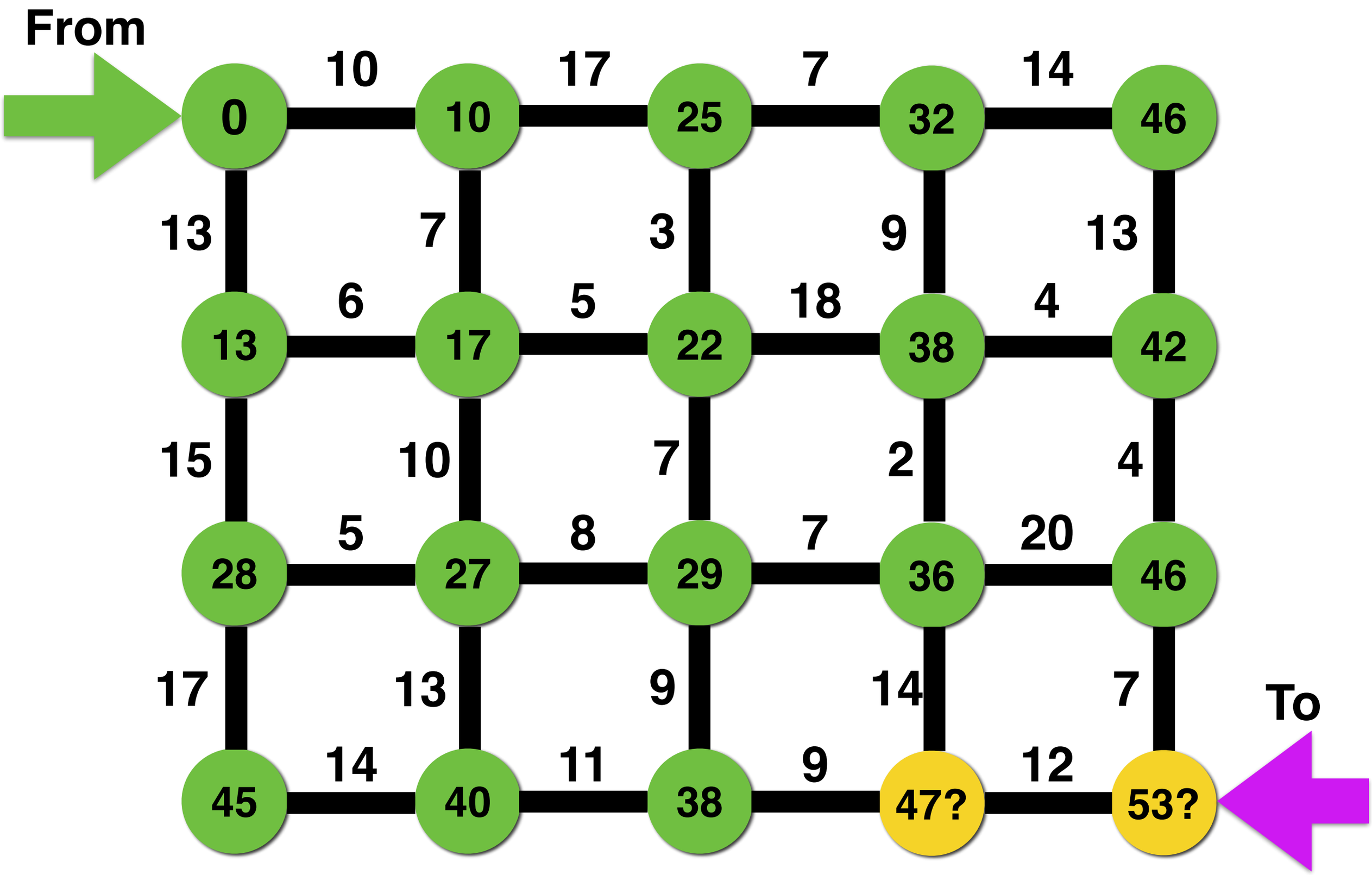

From
0
10
25
32
46?
13
13
6
17
5
22
18
38
4
42
15
10
7
2
4
28
5
27
8
29
7
36
20
56?
17
13
9
14
7
45?
14
40
11
38
9
47?
12
To

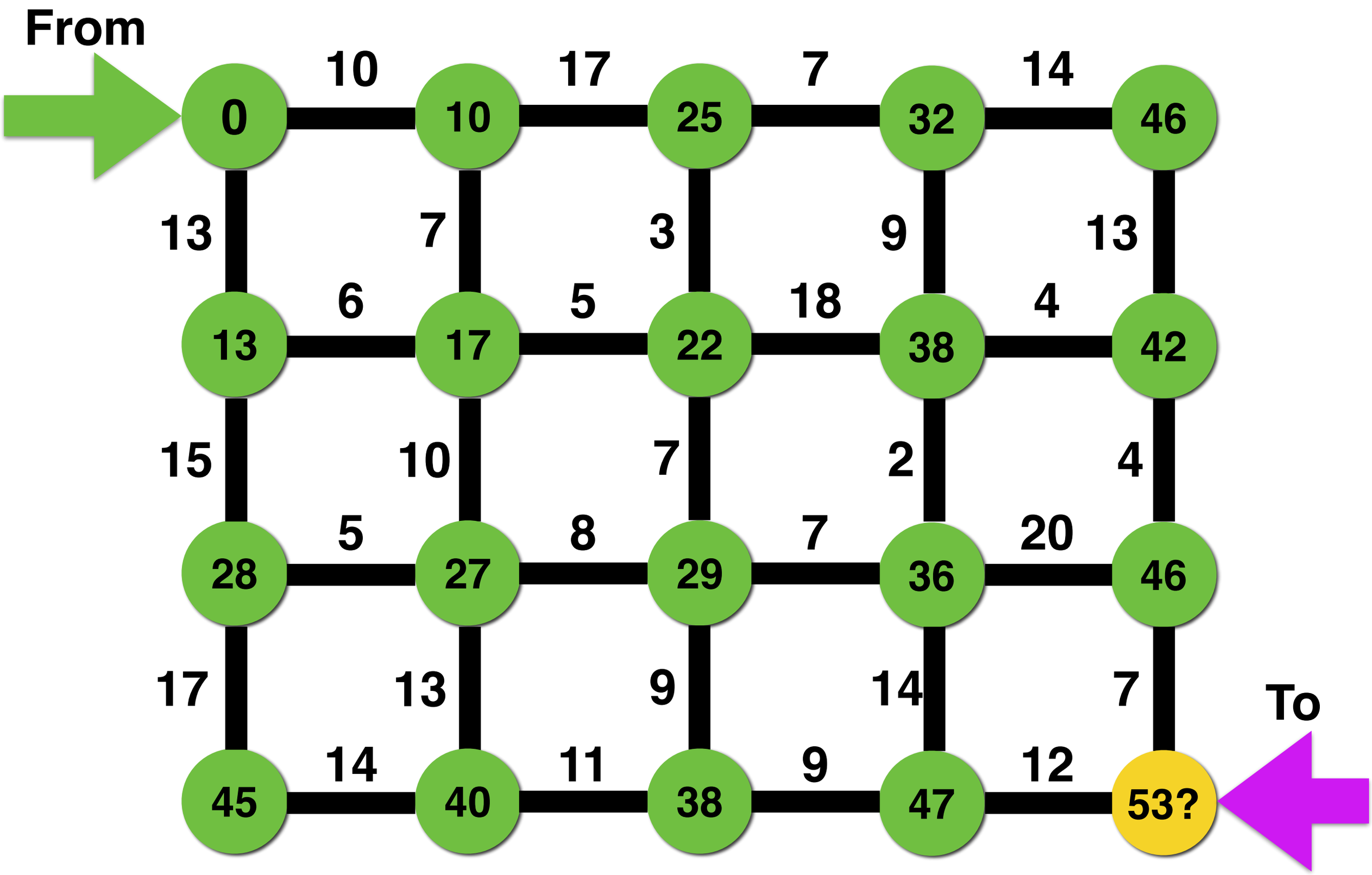

From
0
10
25
32
46?
13
13
6
17
5
22
18
38
4
42
15
10
7
2
4
28
5
27
8
29
7
36
20
46?
17
13
9
14
7
45?
14
40
11
38
9
47?
12
To

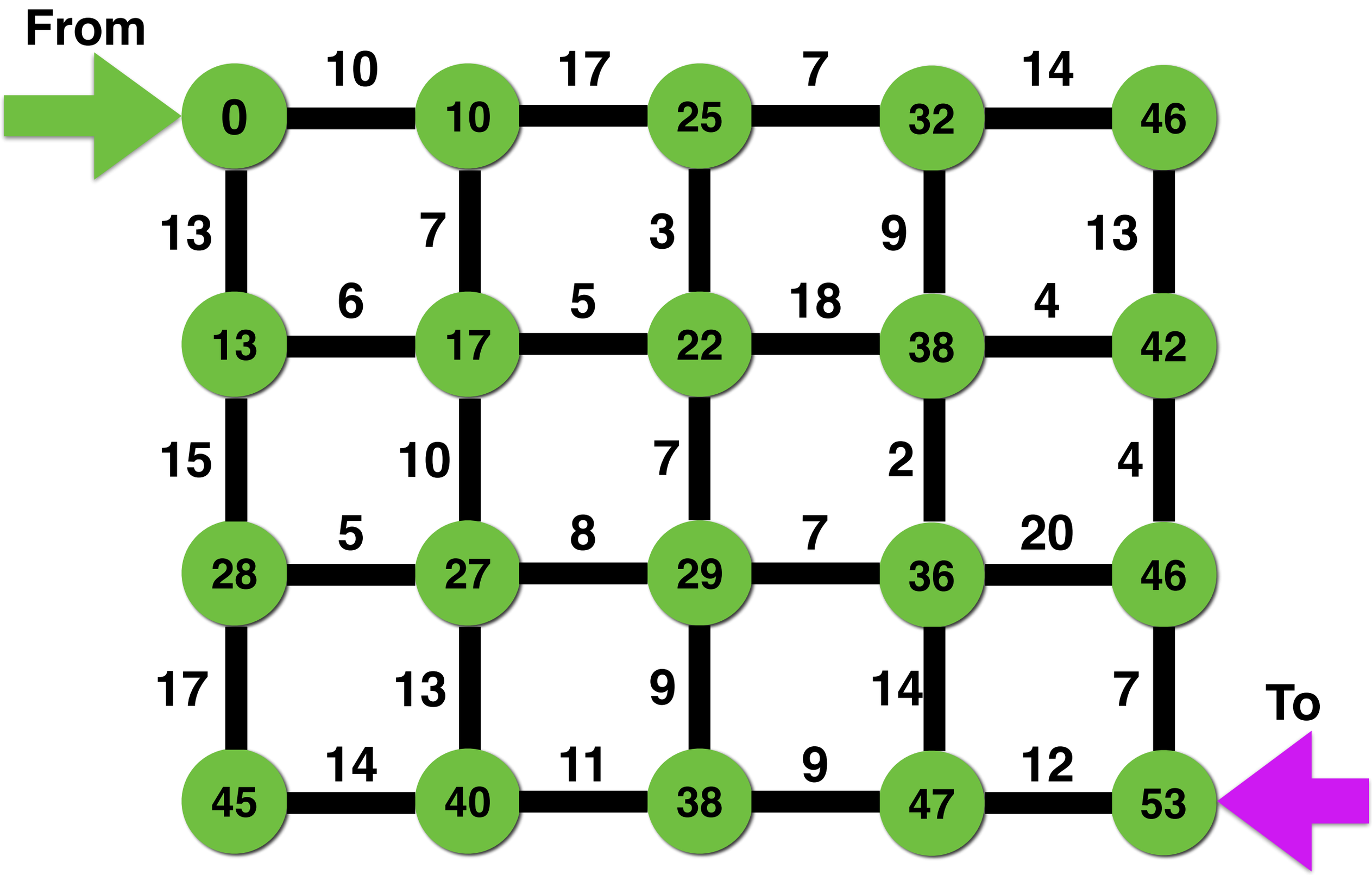

From
0
10
25
32
46?
13
13
6
5
18
4
42
15
10
7
2
4
28
5
27
8
29
7
36
20
46?
17
13
9
14
7
To
45
14
40
11
38
9
47?
12


From
0
10
25
32
46
13
6
5
18
4
13
17
22
38
42
15
10
7
2
4
5
8
7
20
28
27
29
36
46?
17
13
9
14
7
14
11
9
12
45
40
38
47?
To

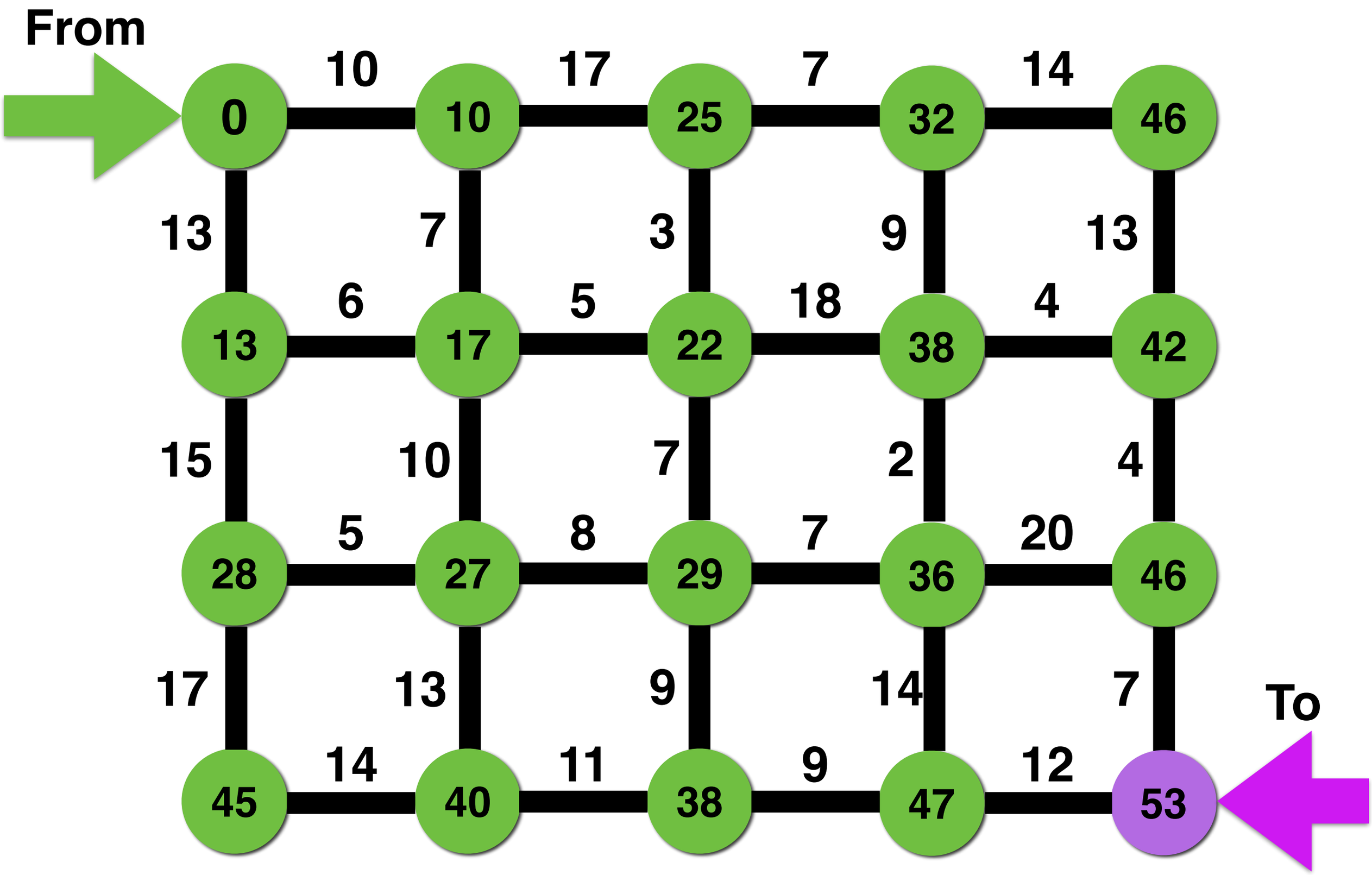

From
0
10
25
32
46
13
6
5
18
4
13
17
22
38
42
15
10
7
2
4
5
8
7
20
28
27
29
36
46
17
13
9
14
7
14
11
9
12
45
40
38
47?
To

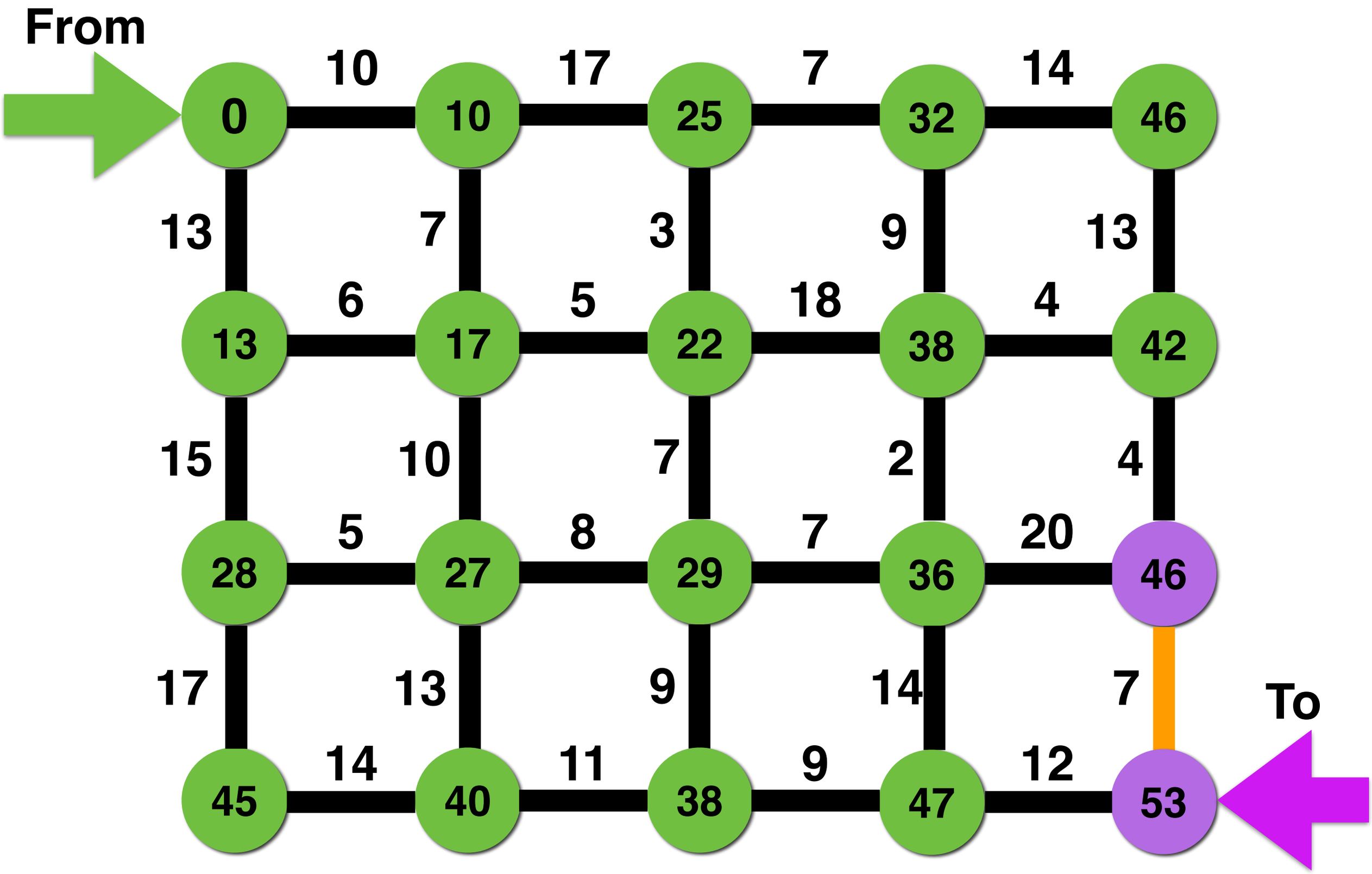

From
0
10
25
32
46
13
6
5
18
4
13
17
22
38
42
15
10
7
2
4
5
8
7
20
28
27
29
36
46
17
13
9
14
7
14
11
9
12
45
40
38
47?
53?
To

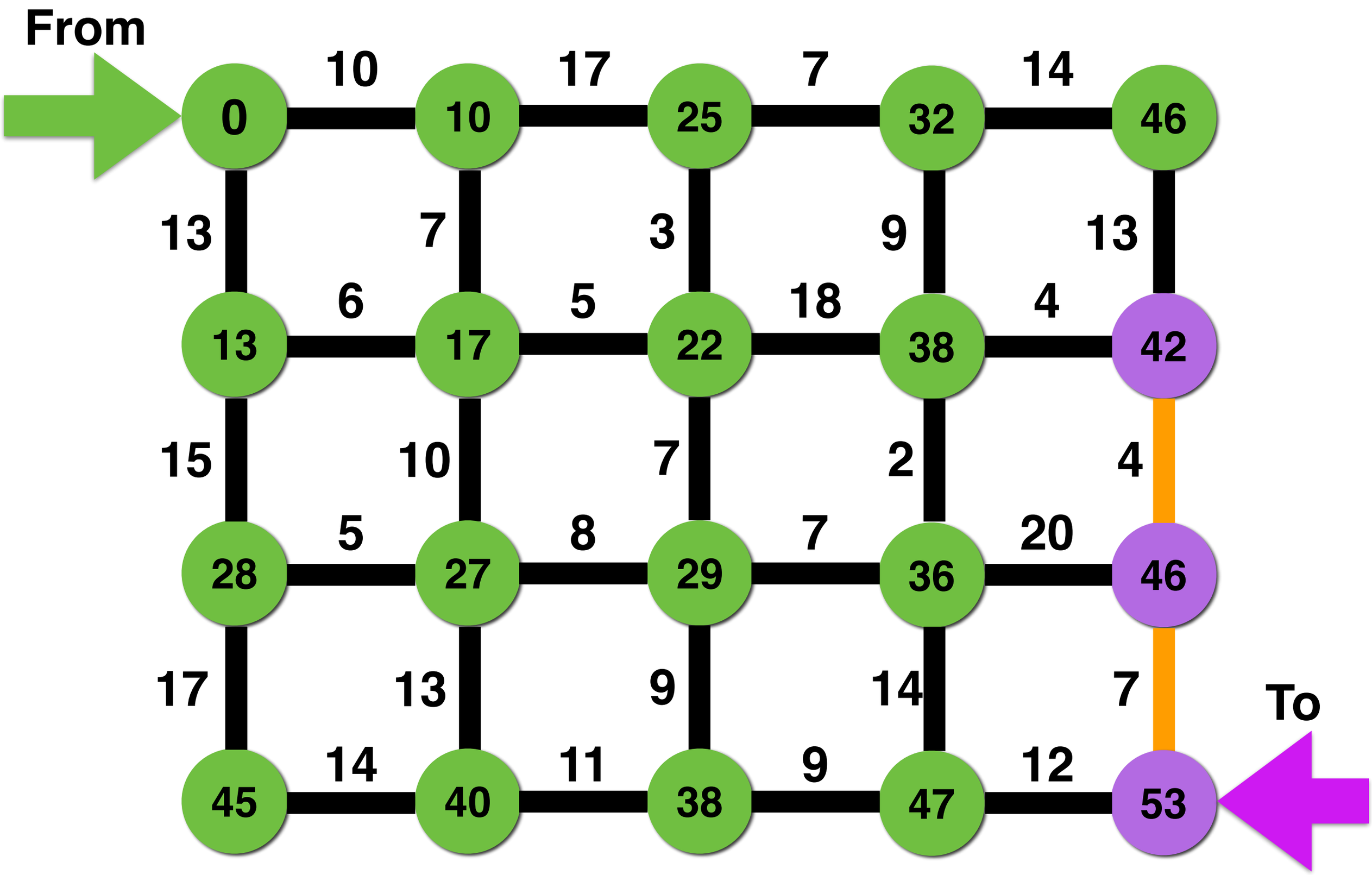

From
0
10
25
32
46
13
6
5
18
4
13
17
22
38
42
15
10
7
2
4
5
8
7
20
28
27
29
36
46
17
13
9
14
7
14
11
9
12
45
40
38
47
53?
To

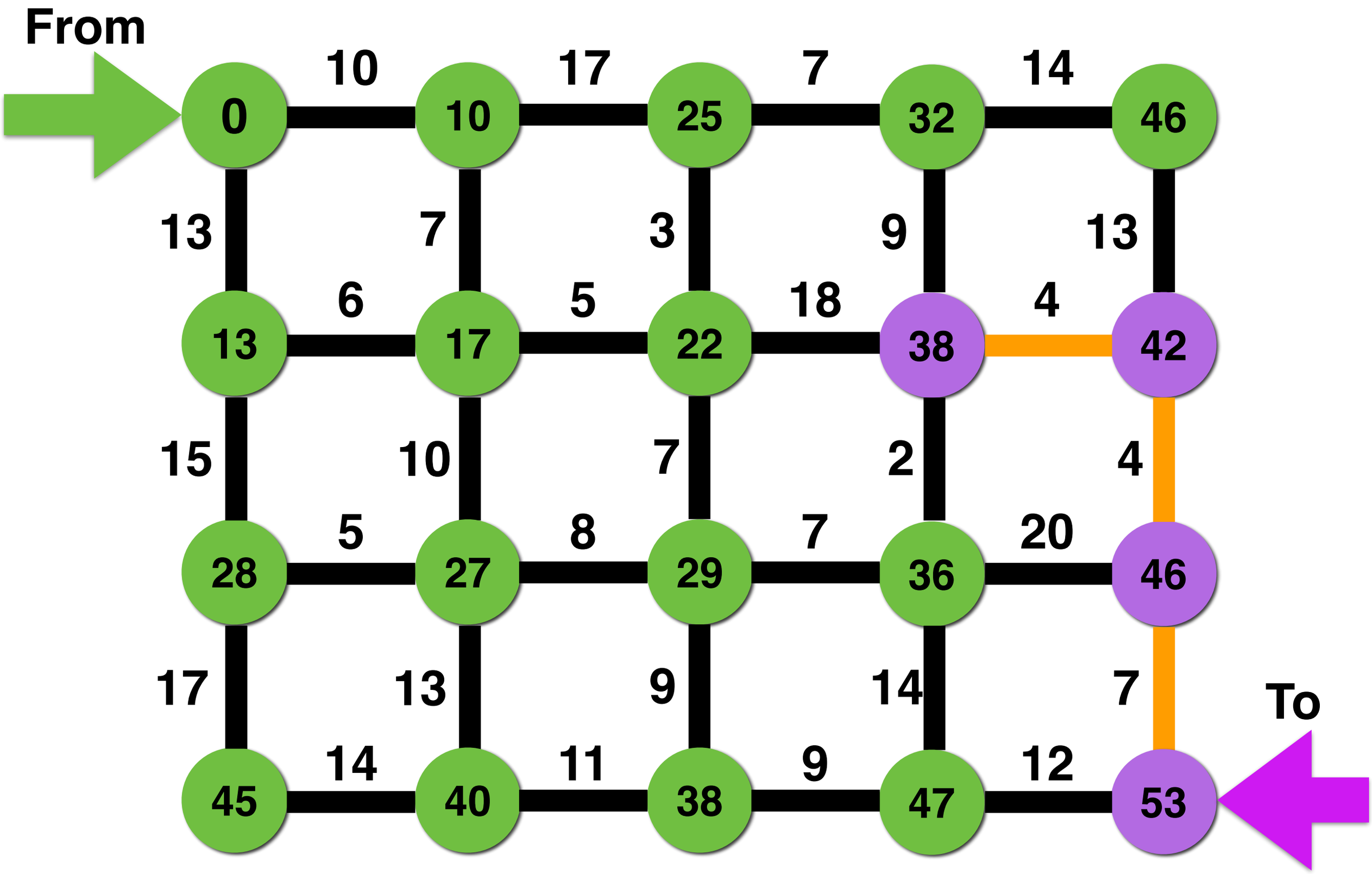

From
0
10
25
32
46
13
6
5
18
4
13
17
22
38
42
15
10
7
2
4
5
8
7
20
28
27
29
36
46
17
13
9
14
7
14
11
9
12
45
40
38
47
53
To

Dijkstra’s Algorithm


From
0
10
25
32
46
13
6
5
18
4
42
15
10
7
2
4
28
5
8
7
20
46
17
13
9
14
7
To
45
14
11
9
12
53


From
0
10
25
32
46
13
6
5
18
4
13
17
22
38
42
15
10
7
2
4
5
8
7
20
28
27
29
36
46
17
13
9
14
7
14
11
9
12
45
40
38
47
53
To


From
0
10
25
32
46
13
6
5
18
4
13
17
22
38
42
15
10
7
2
4
5
8
7
20
28
27
29
36
46
17
13
9
14
7
14
11
9
12
45
40
38
47
53
To


From
0
10
25
32
46
13
6
5
18
4
13
17
22
38
42
15
10
7
2
4
5
8
7
20
28
27
29
36
46
17
13
9
14
7
14
11
9
12
45
40
38
47
53
To

From

0 10 17 7 14 46

13 7 3 9 13

13 6 17 5 22 18 38 4 42

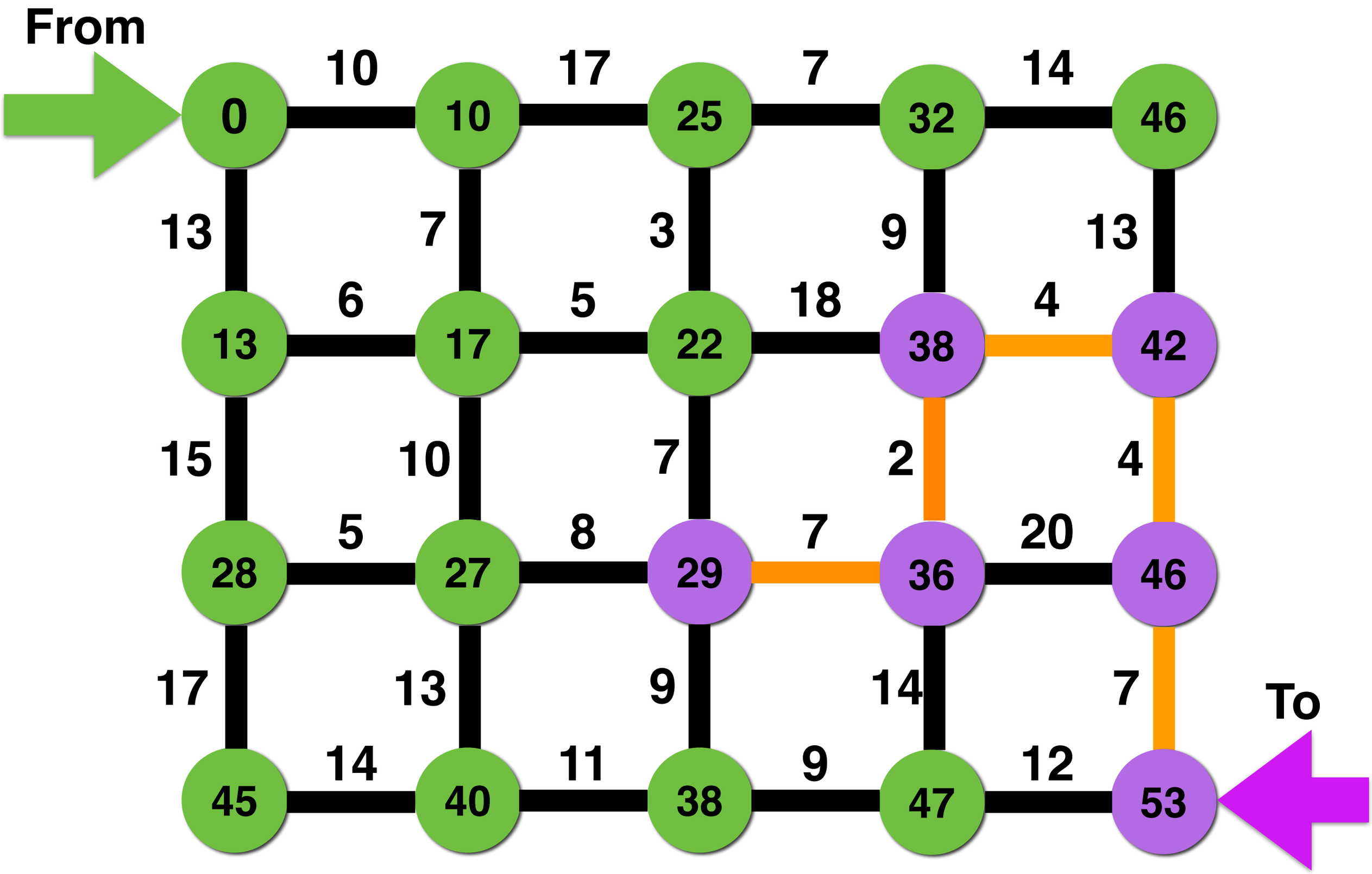
15 10 7 2 4

28 5 27 8 29 7 36 20 46

17 13 9 14 7

45 14 40 11 38 9 47 12 53

To


From
0
10
25
32
46
13
6
5
18
4
38
42
15
2
4
28
5
8
7
20
36
46
17
13
9
14
7
To
45
14
11
9
12
40
38
47
53

From

0 10 17 7 14 46

13 7 3 9 13

13 6 17 5 22 18 38 4 42

15 10 7 2 4

28 5 27 8 29 7 36 20 46

17 13 9 14 7

45 14 40 11 38 9 47 12 53

To

From

0 10 10 17 25 7 32 14 46

13 7 3 9 13

13 6 17 5 22 18 38 4 42

15 10 7 2 4

28 5 27 8 29 7 36 20 46

17 13 9 14 7

45 14 40 11 38 9 47 12 53

To

From

0 10 10 17 25 7 32 14 46

13 7 3 9 13

13 6 17 5 22 18 38 4 42

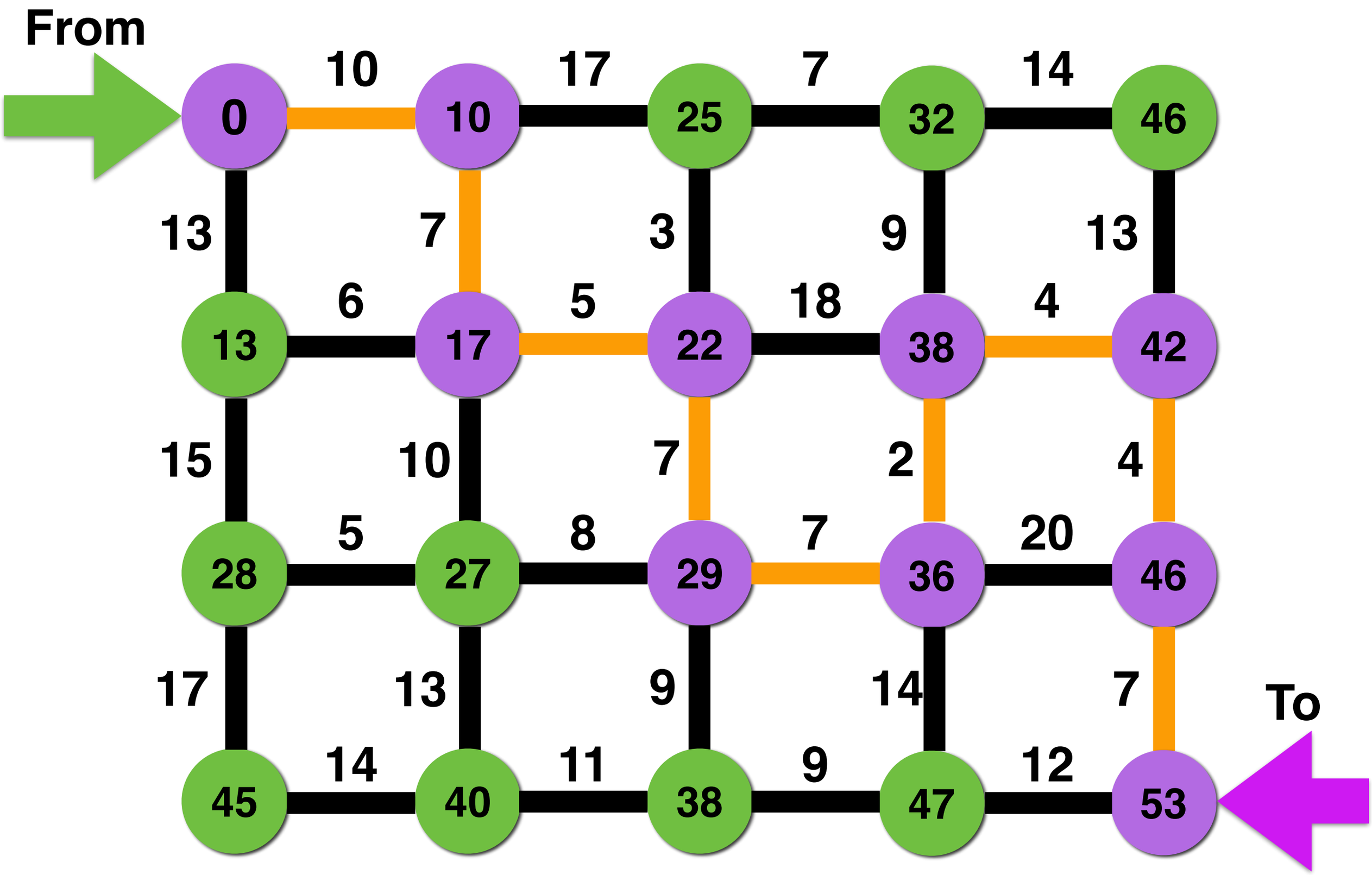
15 10 7 2 4

28 5 27 8 29 7 36 20 46

17 13 9 14 7

45 14 40 11 38 9 47 12 53

To


From
0
10
25
32
46
13
6
5
18
4
42
38
22
17
13
15
10
7
2
4
28
5
8
7
20
46
36
29
27
17
13
9
14
7
45
14
11
9
12
53
40
38
47
To

From

a 10 b 17 c 7 d 14 e

13 7 3 9 13

f 6 g 5 h 18 i 4 j

15 10 7 2 4

k 5 l 8 m 7 n 20 o

17 13 9 14 7

p 14 q 11 r 9 s 12 t

To

Pseudocode

```
dist[s] ←o                                  (distance to source vertex is zero)
for  all v ∈ V–{s}
      do  dist[v] ←∞                        (set all other distances to infinity)
S←∅                                         (S, the set of visited vertices is initially empty)
Q←V                                         (Q, the queue initially contains all vertices)
while Q ≠∅                                  (while the queue is not empty)
do   u ← mindistance(Q,dist)                (select the element of Q with the min. distance)
     S←S∪{u}                                (add u to list of visited vertices)
      for all v ∈ neighbors[u]
            do  if   dist[v] > dist[u] + w(u, v)        (if new shortest path found)
                  then     d[v] ←d[u] + w(u, v)         (set new value of shortest path)
                                                        (if desired, add traceback code)

return dist
```

Reference

(1) http://web.stanford.edu/class/cs106b
(2) http://computer.howstuffworks.com/google-algorithm.htm
(3) http://fortune.com/2012/07/30/amazons-recommendation-secret/
(4) https://en.wikipedia.org/wiki/Dijkstra%27s_algorithm