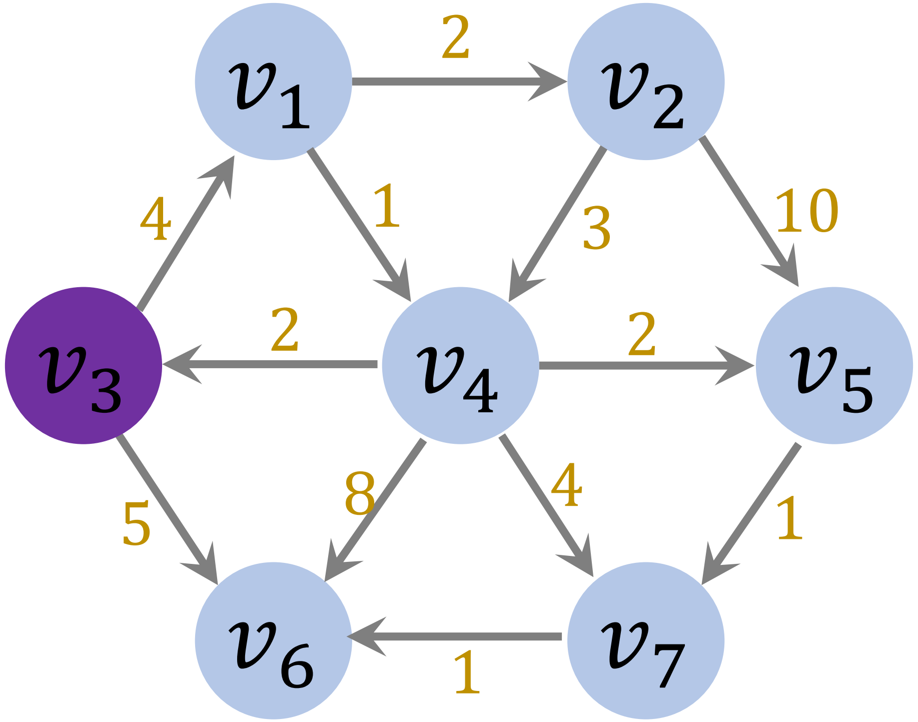
Finding Shortest-Path in Weighted Graphs

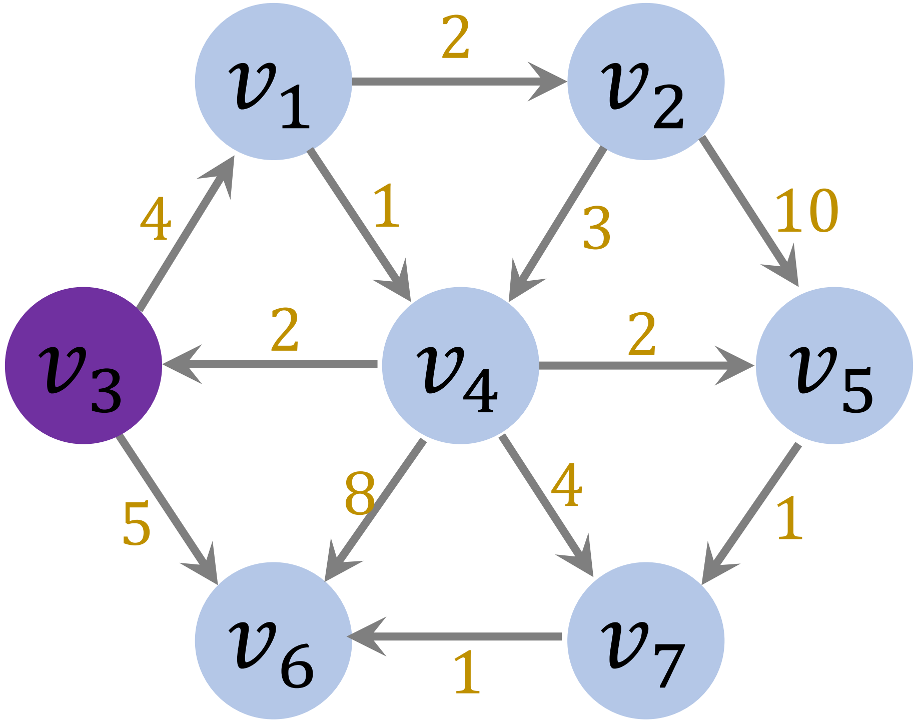
Shusen Wang

http://wangshusen.github.io/

Single-Source Shortest Path in Weighted Graph



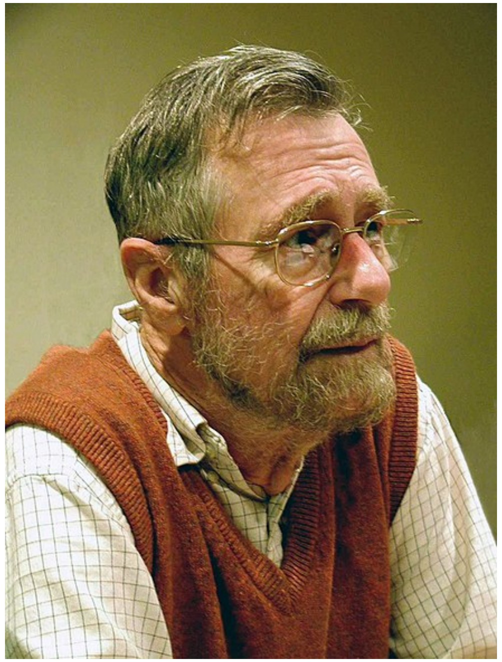
Single-Source Shortest Path in Weighted Graph



vertex	dist	path
v_1	**4**	v_3
v_2	**6**	v_1
v_3	**0**	0
v_4	**5**	v_1
v_5	**7**	v_4
v_6	**5**	v_3
v_7	**9**	0

Dijkstra’s Algorithm

Dijkstra’s Algorithm



Edsger W. Dijkstra

1930 – 2002

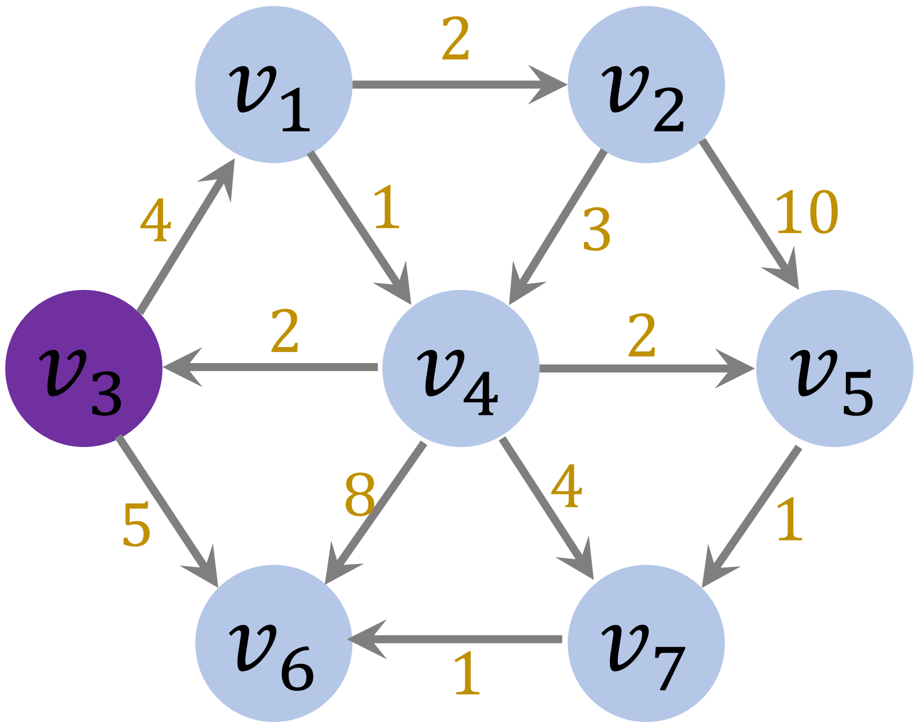
Won Turing Award in 1972

- Dijkstra’s algorithm is for solving the single-source shortest path problem.
- Published in 1959 [1].

Reference

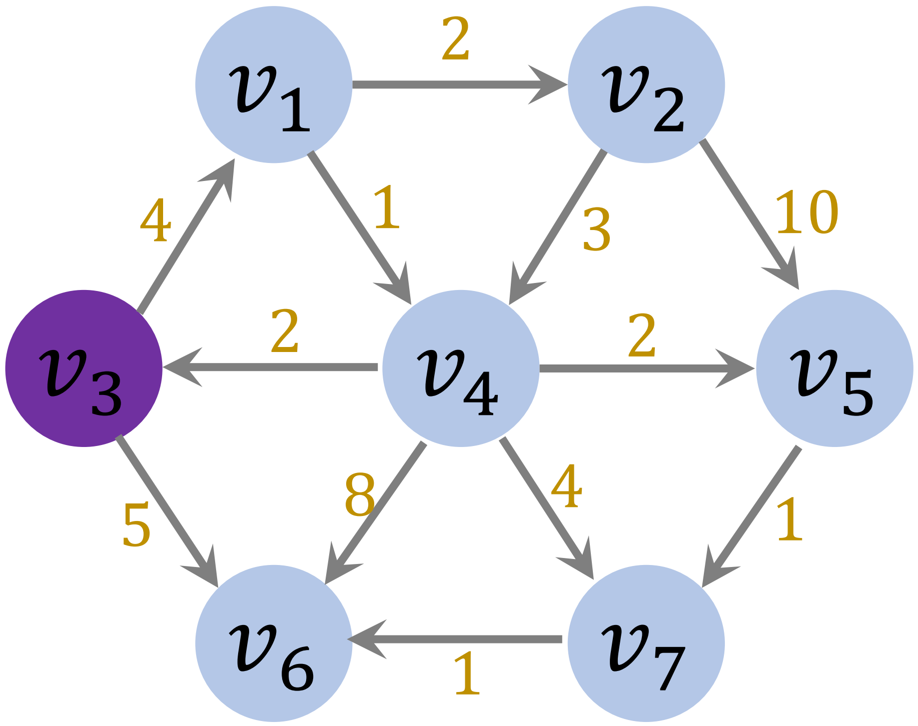
1. E. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*. 1: 269–271, 1959.

Preparations

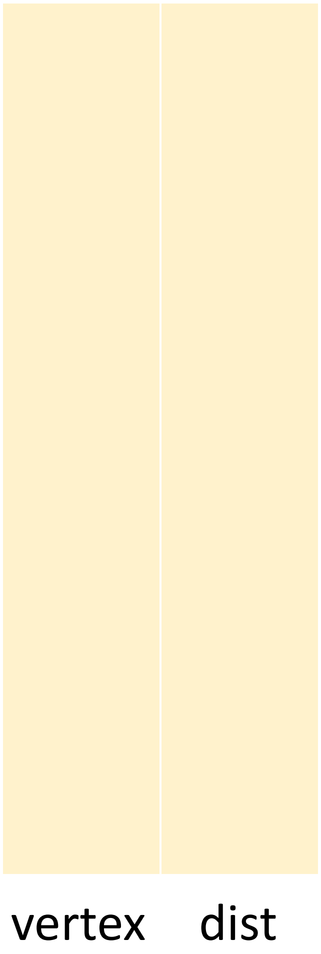


v_3 is the source.

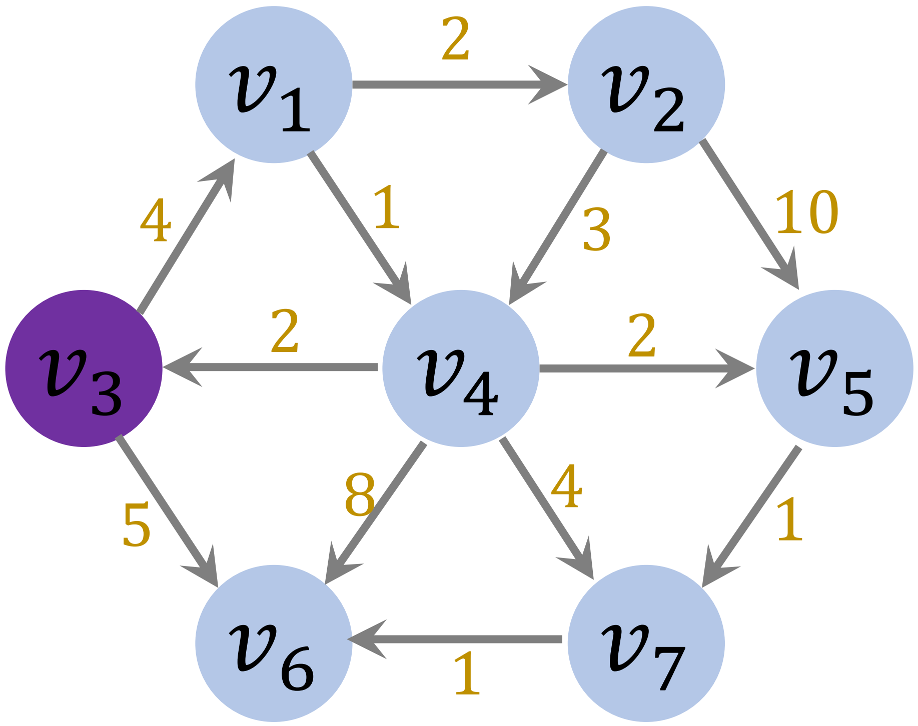
Preparations



Priority Queue:



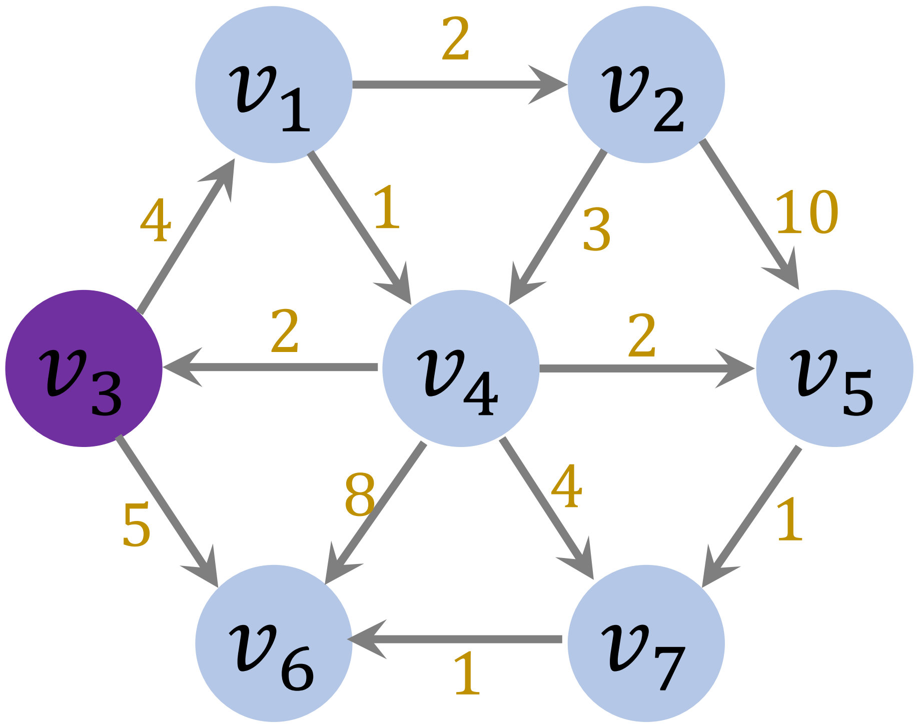
Preparations



Priority Queue:

vertex	dist	path
v_1	∞	
v_2	∞	
v_3	∞	
v_4	∞	
v_5	∞	
v_6	∞	
v_7	∞	

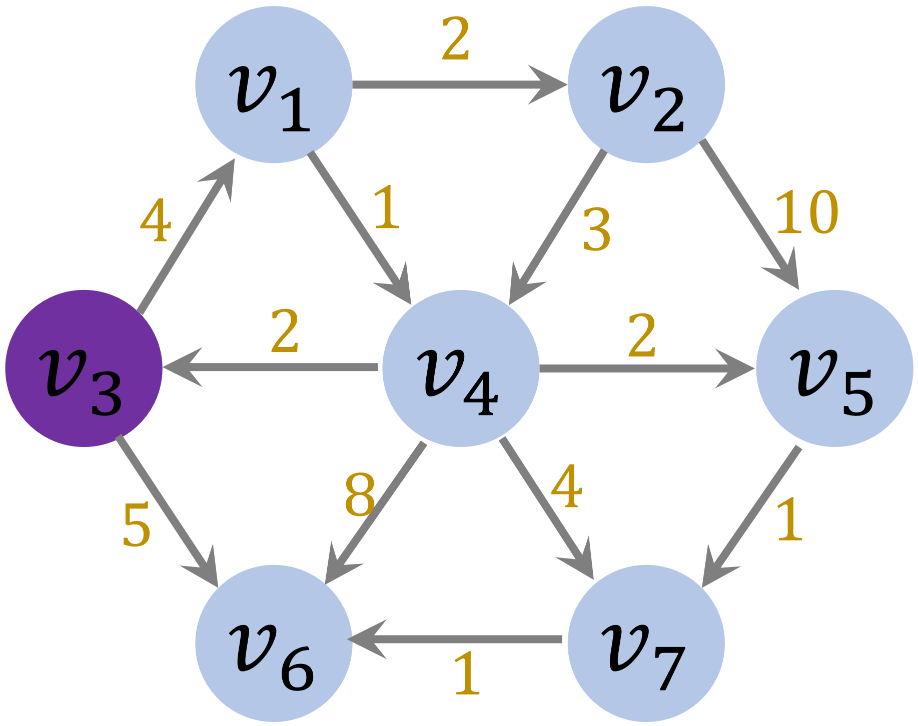
Preparations



Priority Queue:

vertex	dist	path
v_1	∞	0
v_2	∞	0
v_3	∞	0
v_4	∞	0
v_5	∞	0
v_6	∞	0
v_7	∞	0

Initial State

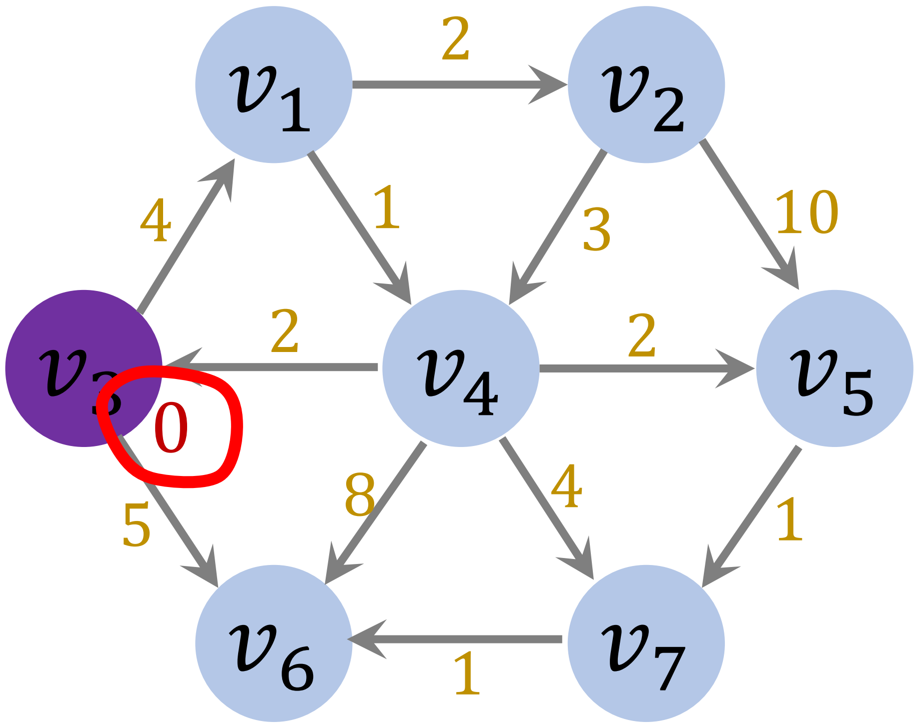


Priority Queue:

vertex	dist

vertex	dist	path
v_1	∞	0
v_2	∞	0
v_3	**0**	0
v_4	∞	0
v_5	∞	0
v_6	∞	0
v_7	∞	0

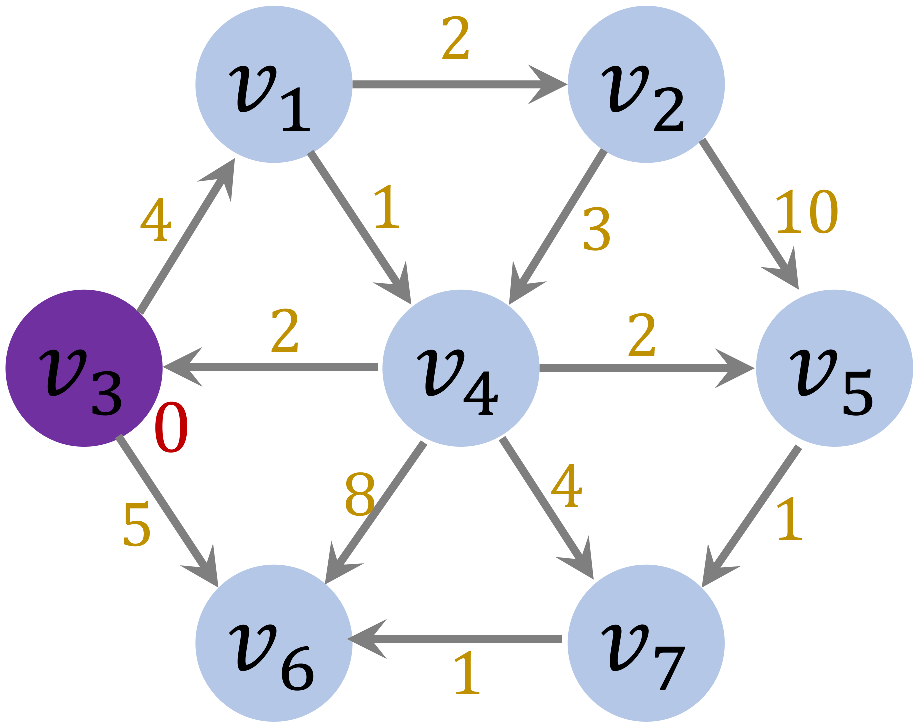
Initial State



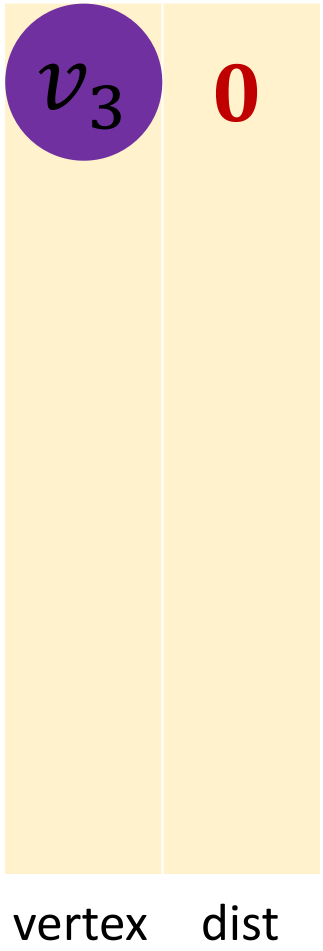
Priority Queue:

vertex	dist	path
v_1	∞	0
v_2	∞	0
v_3	**0**	0
v_4	∞	0
v_5	∞	0
v_6	∞	0
v_7	∞	0

Initial State

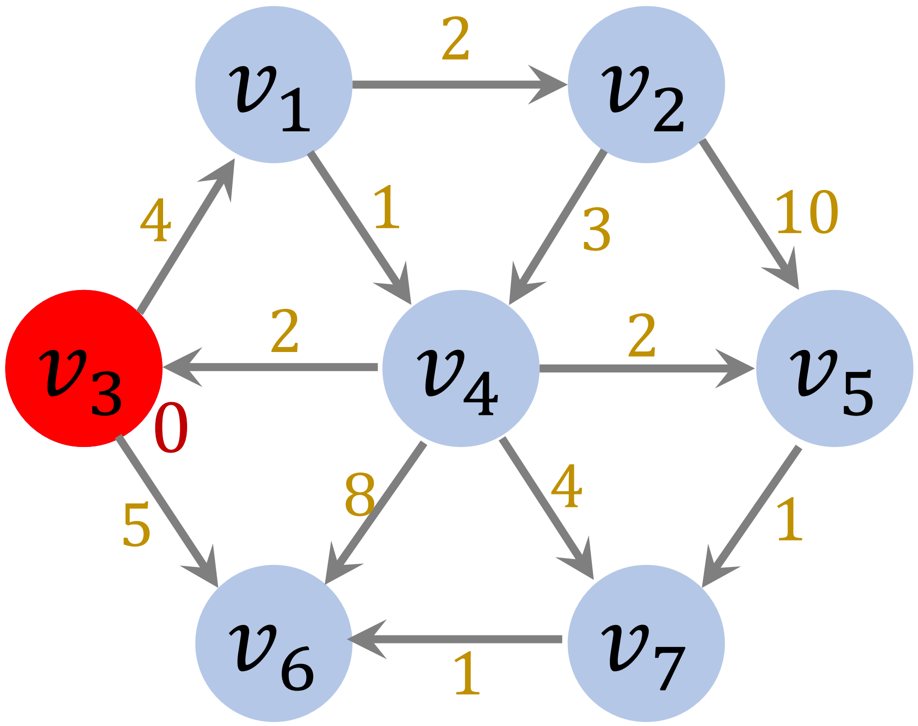


Priority Queue:



vertex	dist	path
v_1	∞	0
v_2	∞	0
v_3	**0**	0
v_4	∞	0
v_5	∞	0
v_6	∞	0
v_7	∞	0

Iteration 1

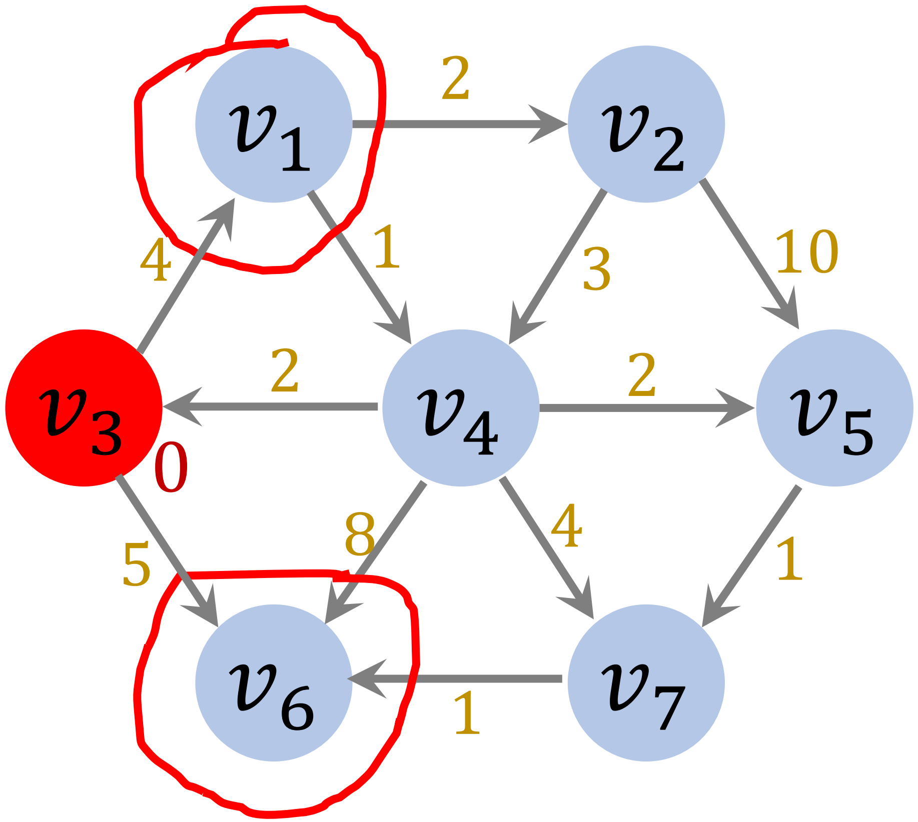


Priority Queue:

vertex	dist
v_3	**0**

vertex	dist	path
v_1	∞	0
v_2	∞	0
v_3	**0**	0
v_4	∞	0
v_5	∞	0
v_6	∞	0
v_7	∞	0

Iteration 1

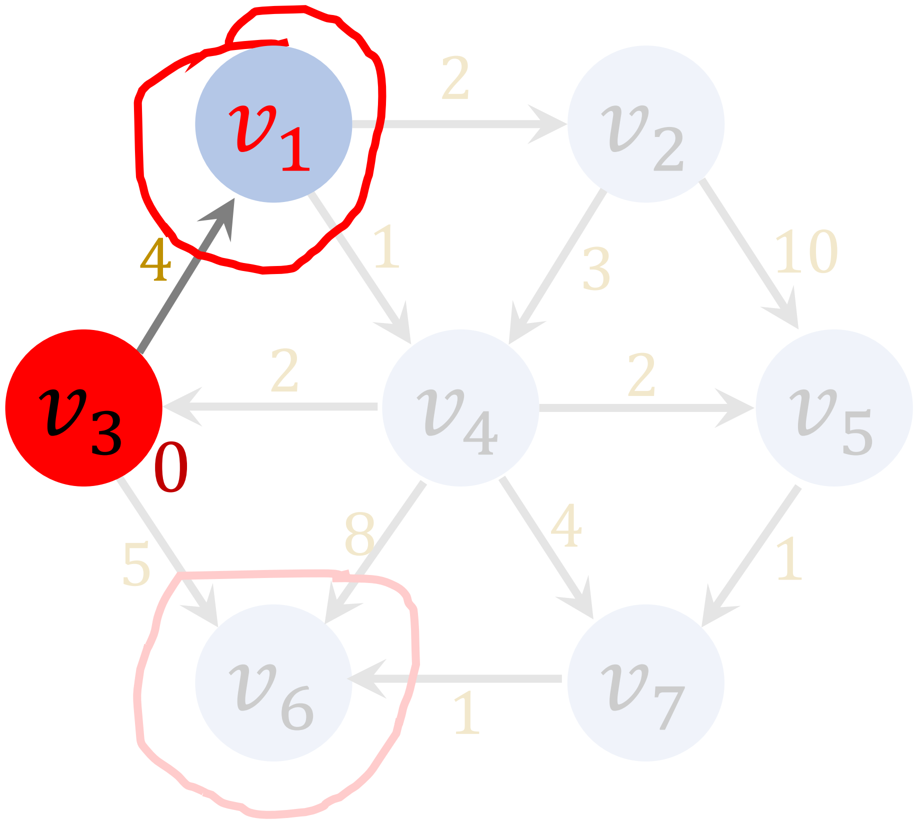


Priority Queue:

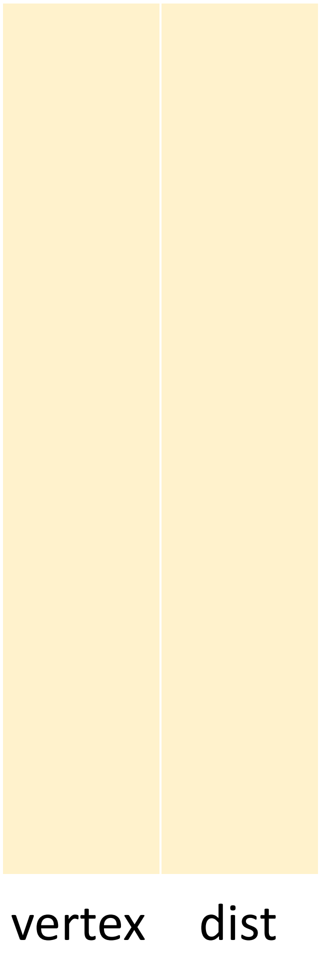
vertex	dist

vertex	dist	path
v_1	∞	0
v_2	∞	0
v_3	**0**	0
v_4	∞	0
v_5	∞	0
v_6	∞	0
v_7	∞	0

Iteration 1(A)

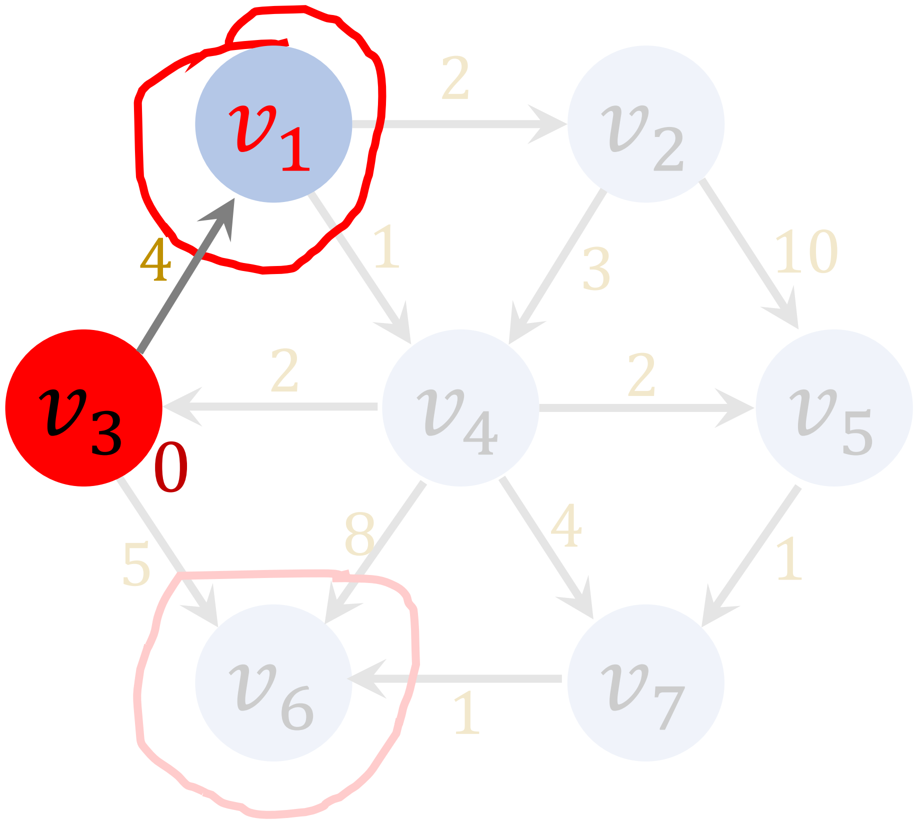


Priority Queue:



vertex	dist	path
v_1	∞	0
v_2	∞	0
v_3	**0**	0
v_4	∞	0
v_5	∞	0
v_6	∞	0
v_7	∞	0

Iteration 1(A)

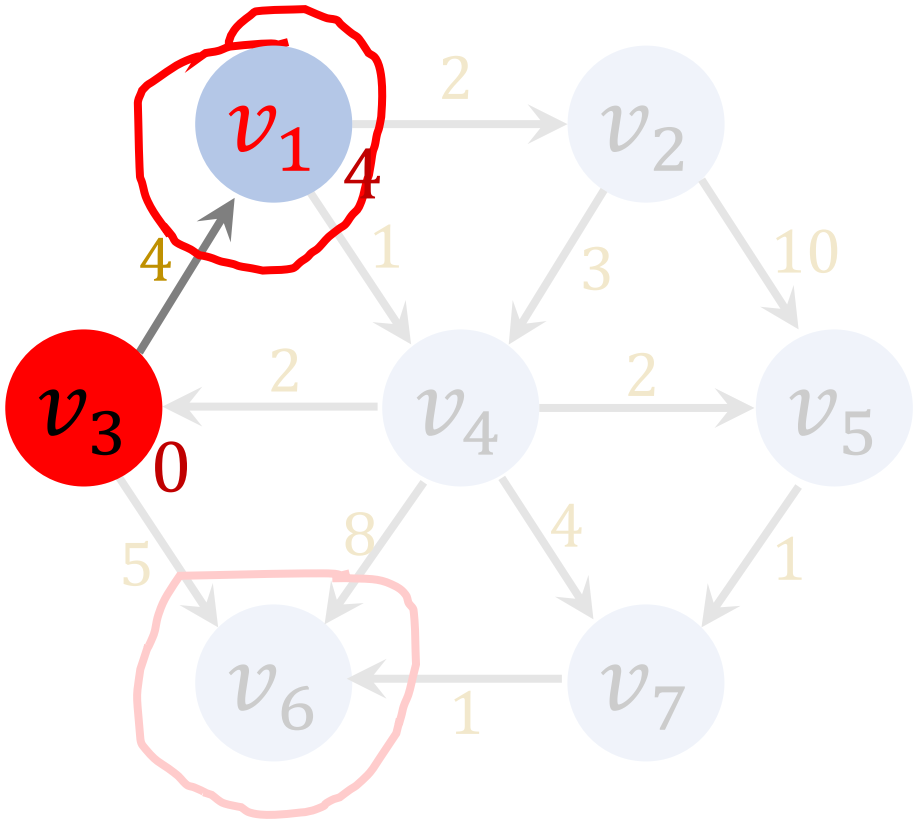


Priority Queue:

vertex	dist

vertex	dist	path
v_1	∞	0
v_2	∞	0
v_3	**0**	0
v_4	∞	0
v_5	∞	0
v_6	∞	0
v_7	∞	0

Iteration 1(A)

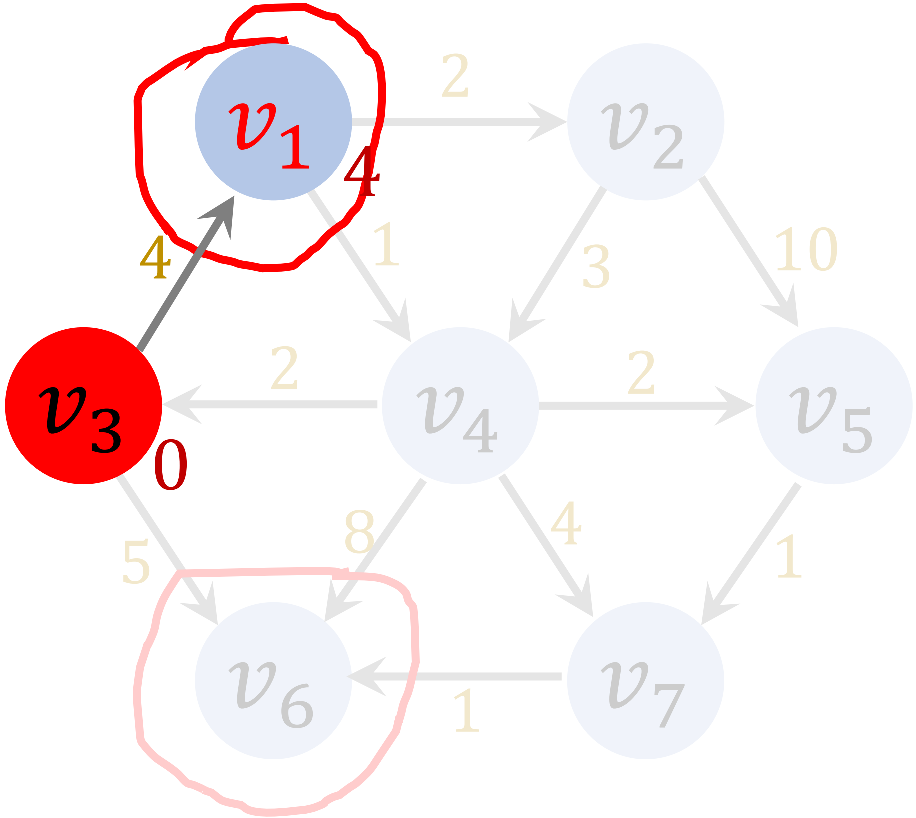


Priority Queue:

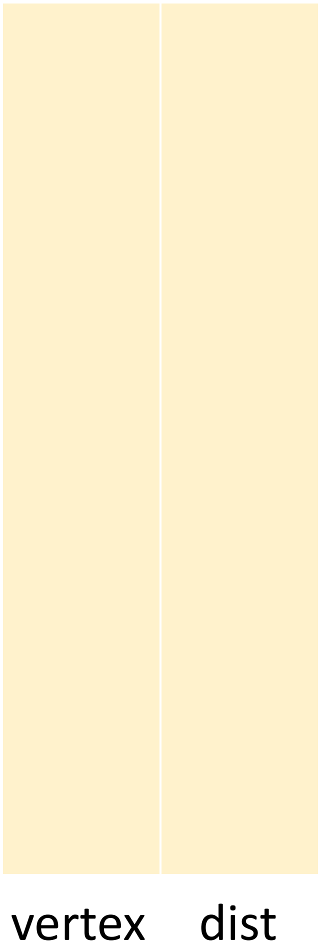
vertex	dist

vertex	dist	path
v_1	4	0
v_2	∞	0
v_3	**0**	0
v_4	∞	0
v_5	∞	0
v_6	∞	0
v_7	∞	0

Iteration 1(A)

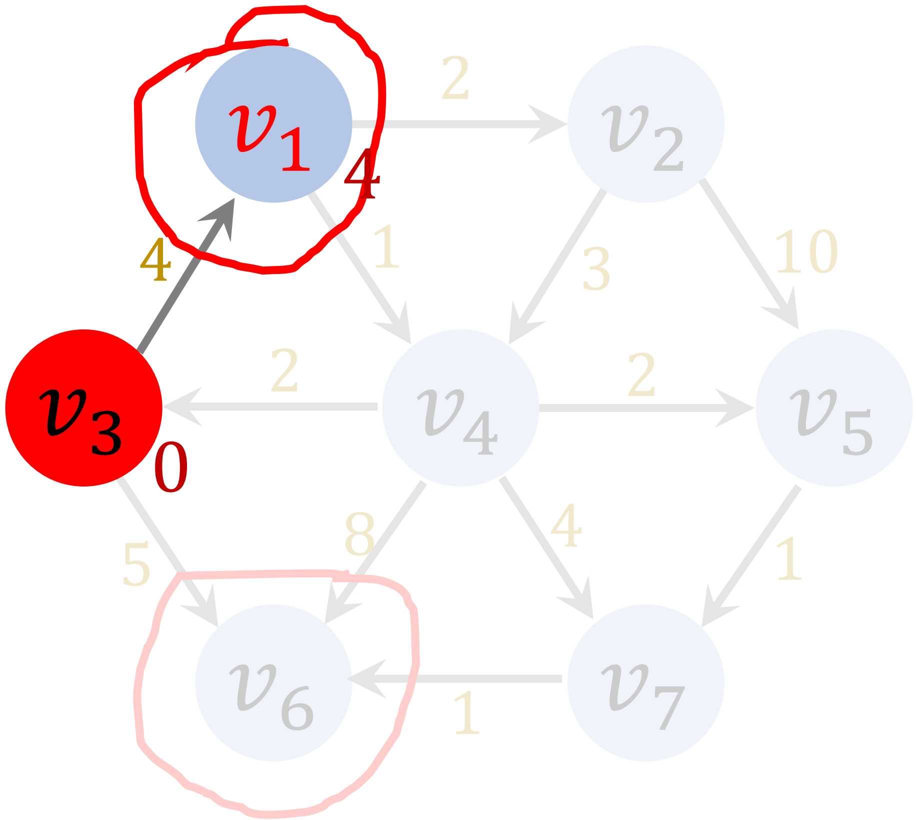


Priority Queue:



vertex	dist	path
v_1	**4**	v_3
v_2	∞	0
v_3	**0**	0
v_4	∞	0
v_5	∞	0
v_6	∞	0
v_7	∞	0

Iteration 1(A)

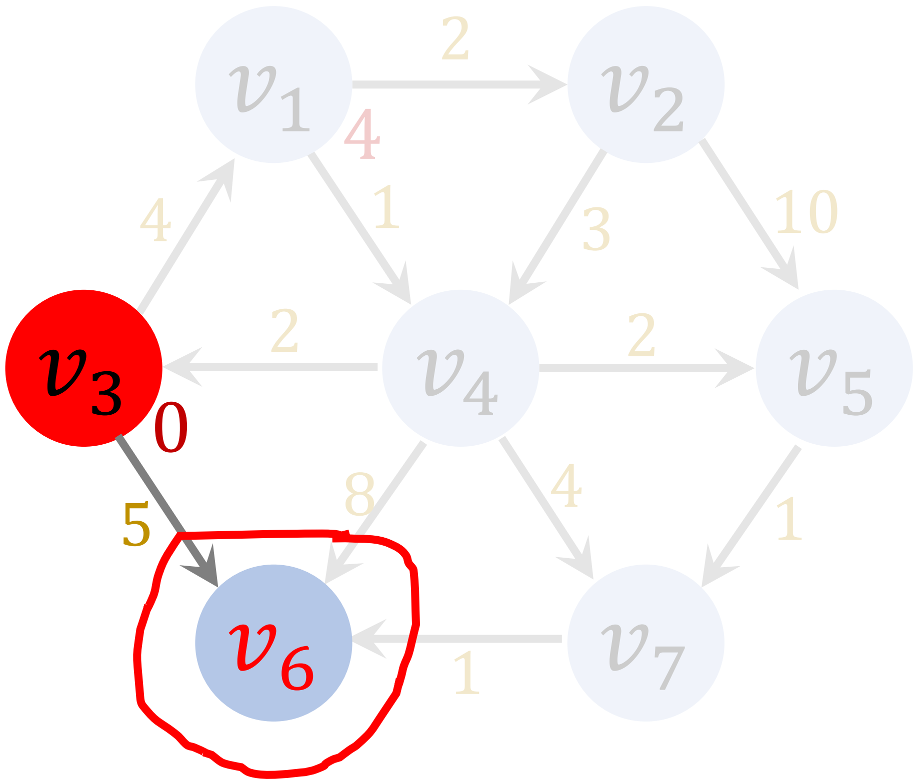


Priority Queue:

vertex	dist
v_1	**4**

vertex	dist	path
v_1	**4**	v_3
v_2	∞	0
v_3	**0**	0
v_4	∞	0
v_5	∞	0
v_6	∞	0
v_7	∞	0

Iteration 1(B)

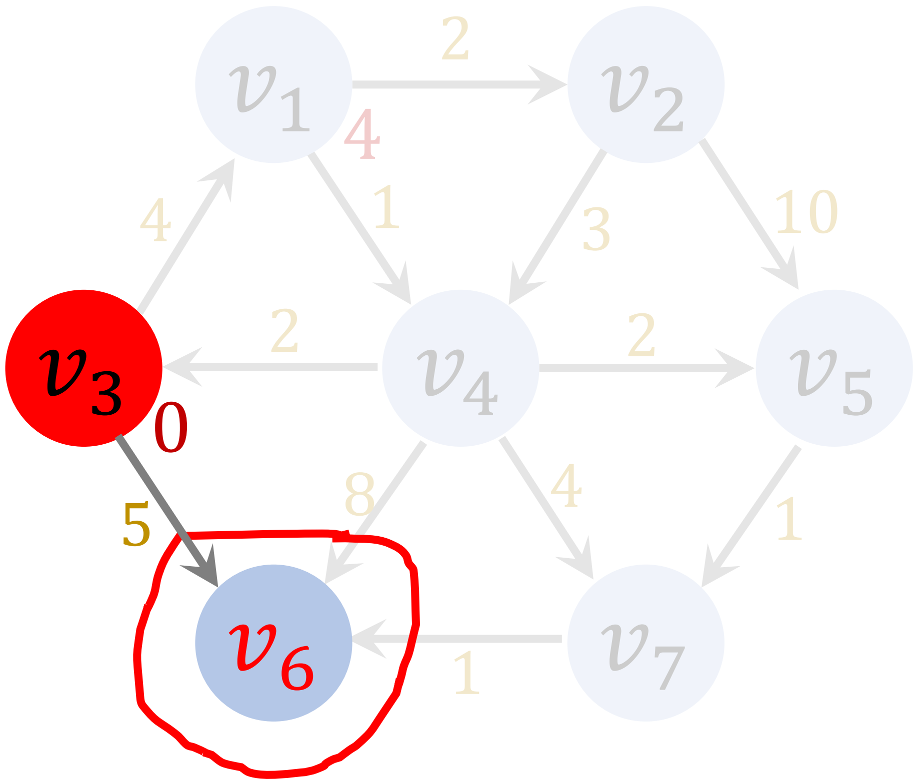


Priority Queue:

vertex	dist
v_1	4

vertex	dist	path
v_1	4	v_3
v_2	∞	0
v_3	0	0
v_4	∞	0
v_5	∞	0
v_6	∞	0
v_7	∞	0

Iteration 1(B)

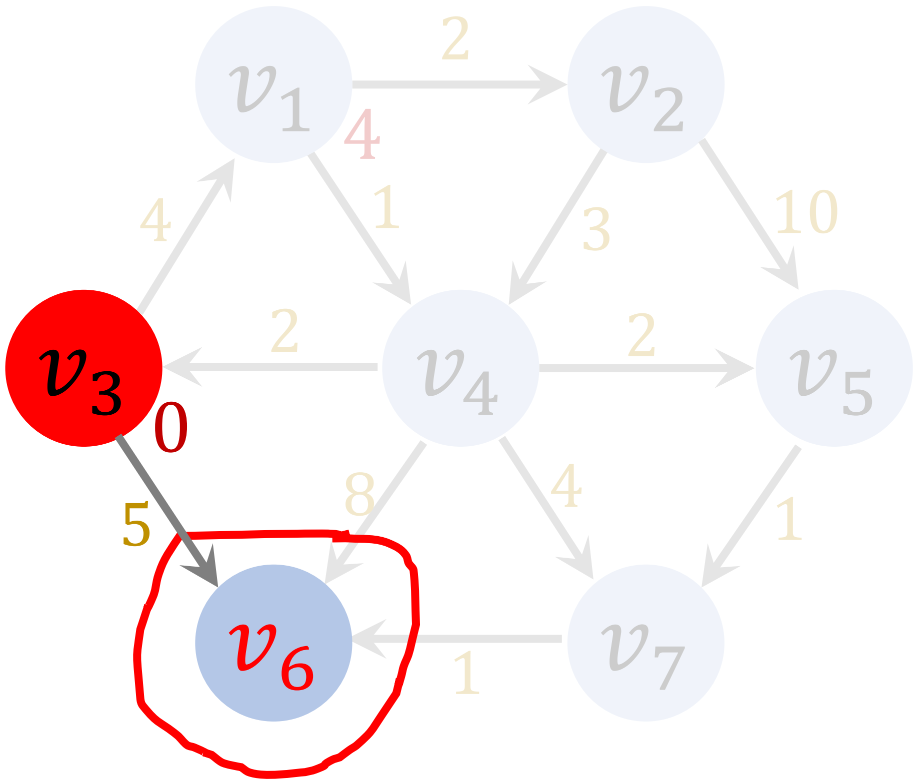


Priority Queue:

vertex	dist
v_1	**4**

vertex	dist	path
v_1	**4**	v_3
v_2	∞	0
v_3	**0**	0
v_4	∞	0
v_5	∞	0
v_6	∞	0
v_7	∞	0

Iteration 1(B)

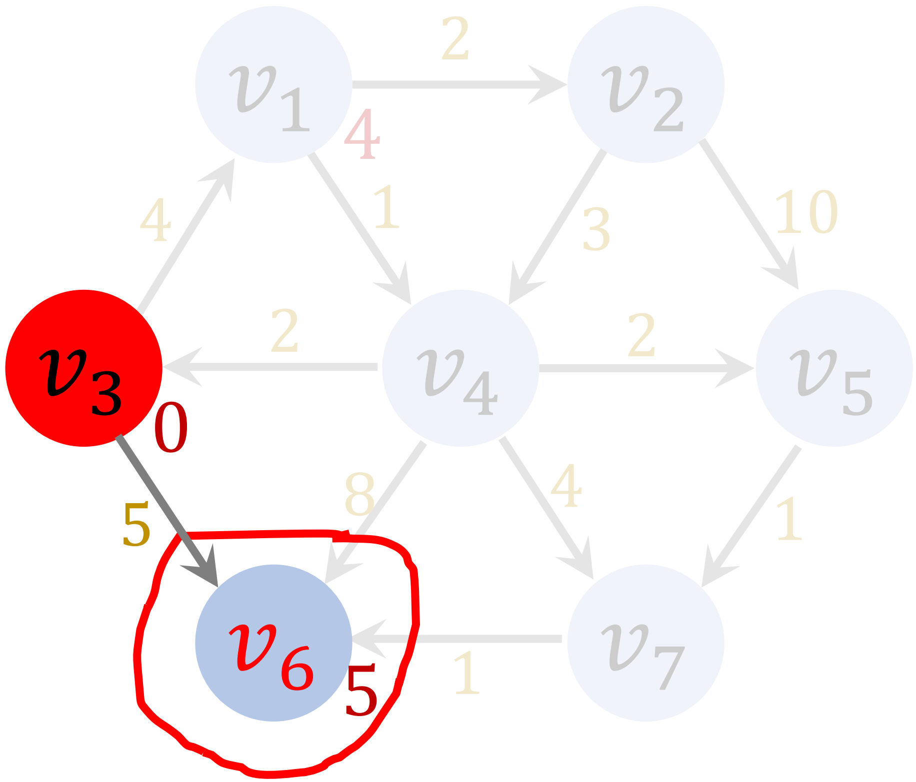


Priority Queue:

vertex	dist
v_1	4

vertex	dist	path
v_1	4	v_3
v_2	∞	0
v_3	0	0
v_4	∞	0
v_5	∞	0
v_6	∞	0
v_7	∞	0

Iteration 1(B)

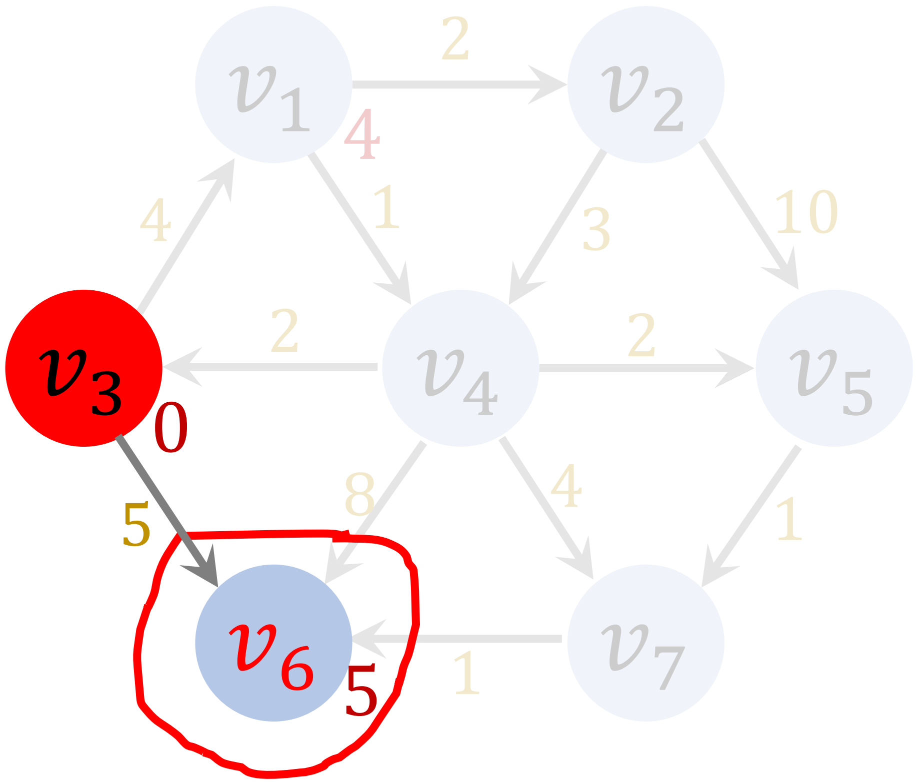


Priority Queue:

vertex	dist
v_1	**4**

vertex	dist	path
v_1	**4**	v_3
v_2	∞	0
v_3	**0**	0
v_4	∞	0
v_5	∞	0
v_6	**5**	0
v_7	∞	0

Iteration 1(B)

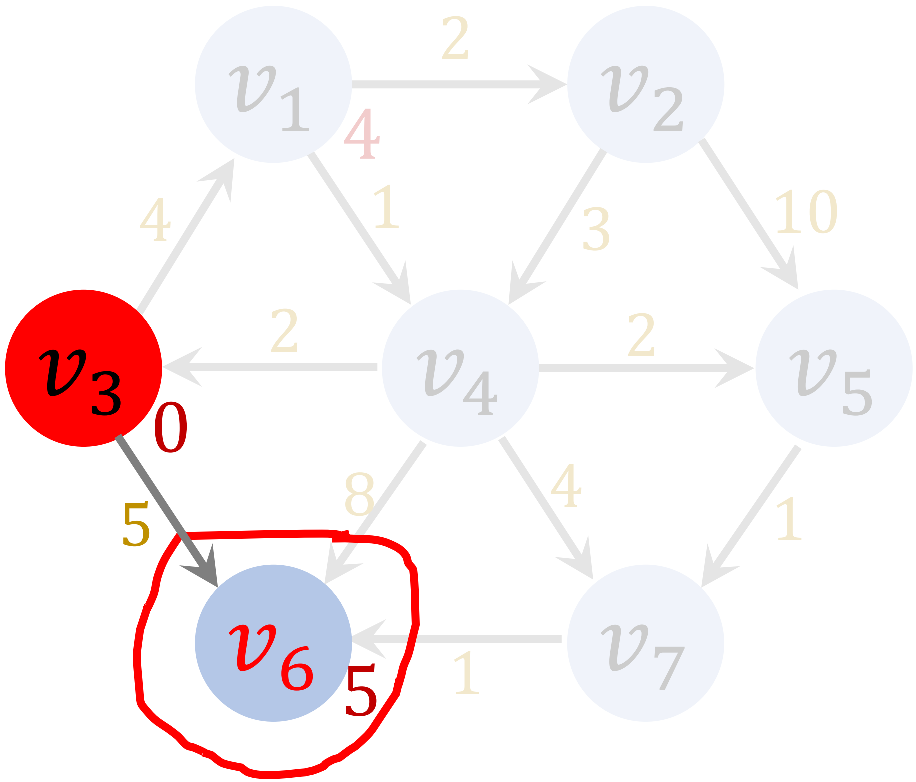


Priority Queue:

vertex	dist
v_1	**4**

vertex	dist	path
v_1	**4**	v_3
v_2	∞	0
v_3	**0**	0
v_4	∞	0
v_5	∞	0
v_6	**5**	v_3
v_7	∞	0

Iteration 1(B)

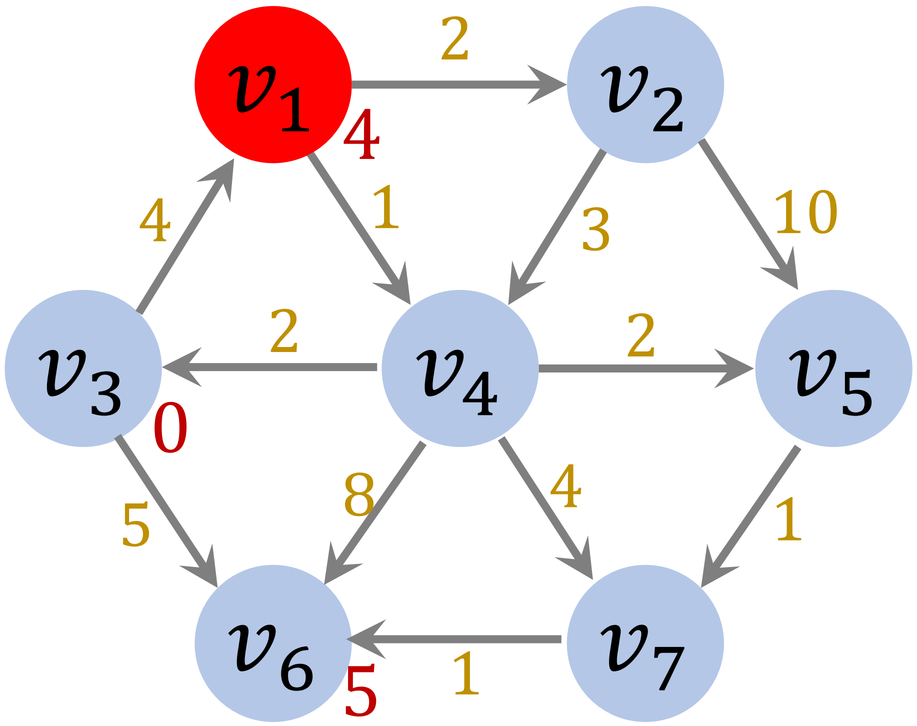


Priority Queue:

vertex	dist
v_1	**4**
v_6	**5**

vertex	dist	path
v_1	**4**	v_3
v_2	∞	0
v_3	**0**	0
v_4	∞	0
v_5	∞	0
v_6	**5**	v_3
v_7	∞	0

Iteration 2

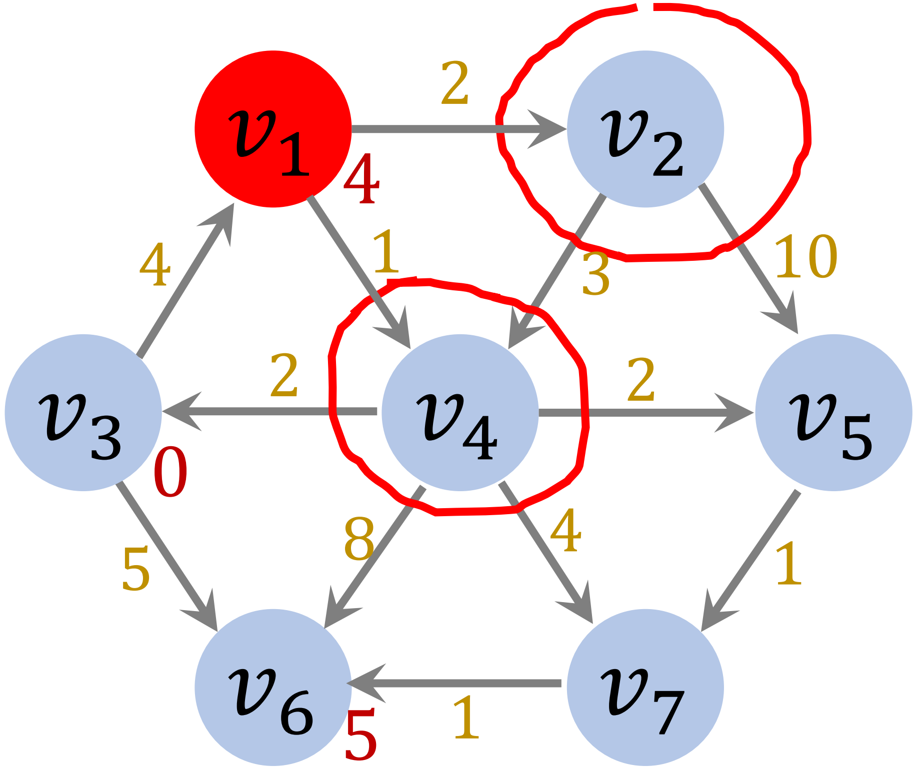


Priority Queue:

vertex	dist
v_1	**4**
v_6	**5**

vertex	dist	path
v_1	**4**	v_3
v_2	∞	0
v_3	**0**	0
v_4	∞	0
v_5	∞	0
v_6	**5**	v_3
v_7	∞	0

Iteration 2

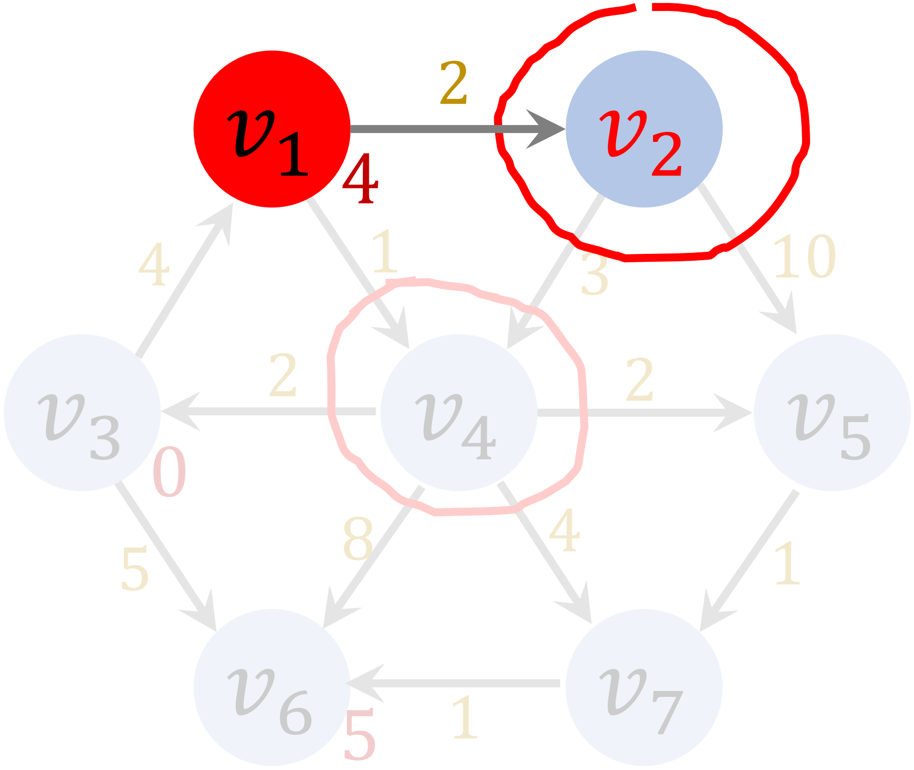


Priority Queue:

vertex	dist
v_6	**5**

vertex	dist	path
v_1	**4**	v_3
v_2	∞	0
v_3	**0**	0
v_4	∞	0
v_5	∞	0
v_6	**5**	v_3
v_7	∞	0

Iteration 2(A)

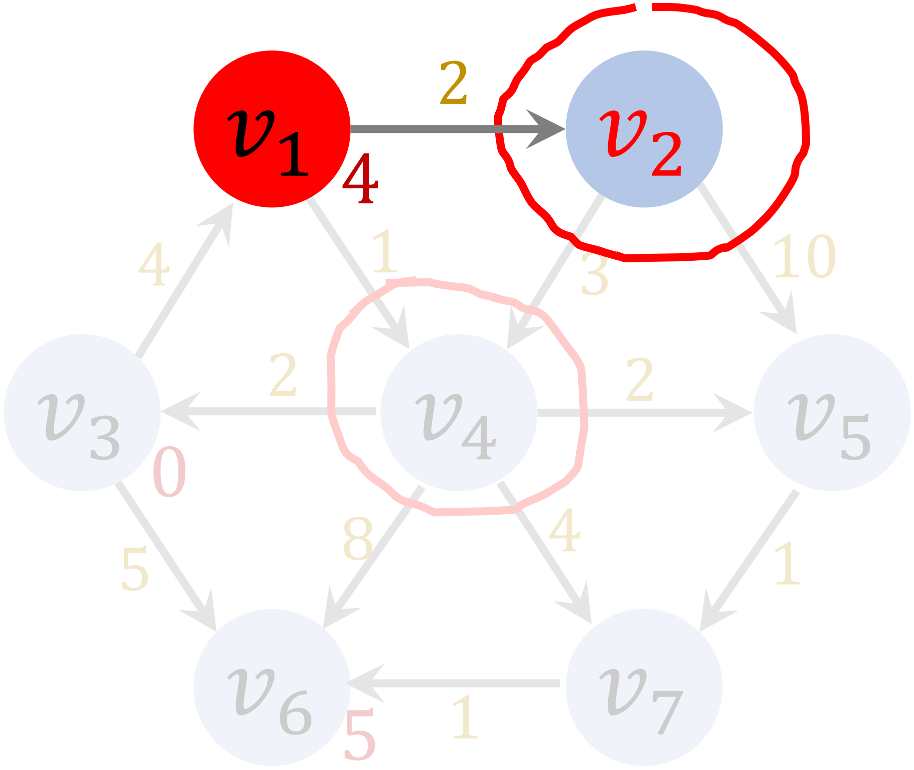


Priority Queue:

vertex	dist
v_6	**5**

vertex	dist	path
v_1	**4**	v_3
v_2	∞	0
v_3	**0**	0
v_4	∞	0
v_5	∞	0
v_6	**5**	v_3
v_7	∞	0

Iteration 2(A)



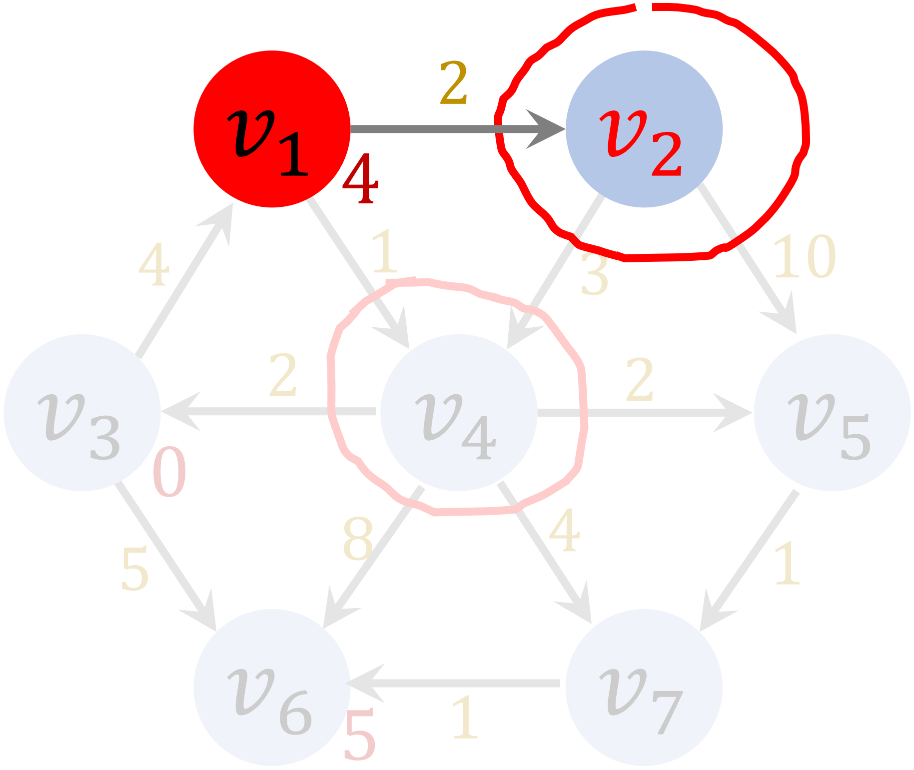
$$d_{\text{new}} = 4 + 2 = 6.$$

Priority Queue:

vertex	dist
v_6	**5**

vertex	dist	path
v_1	**4**	v_3
v_2	∞	0
v_3	**0**	0
v_4	∞	0
v_5	∞	0
v_6	**5**	v_3
v_7	∞	0

Iteration 2(A)



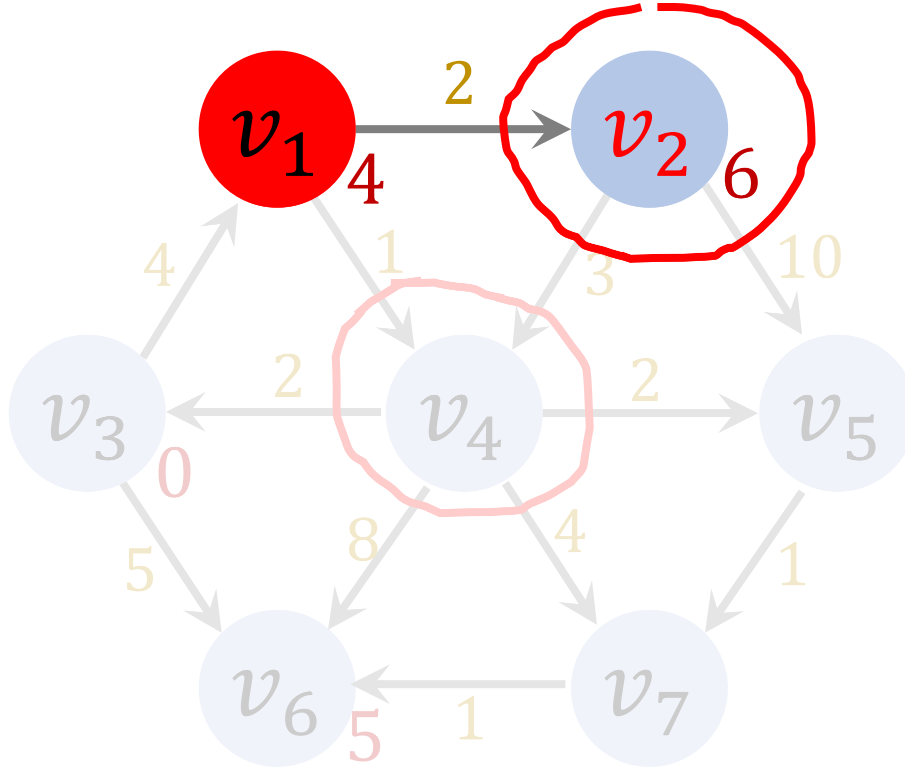
$d_{\text{new}} = 4 + 2 = 6.$

Priority Queue:

vertex	dist
v_6	**5**

vertex	dist	path
v_1	**4**	v_3
v_2	∞	0
v_3	**0**	0
v_4	∞	0
v_5	∞	0
v_6	**5**	v_3
v_7	∞	0

Iteration 2(A)



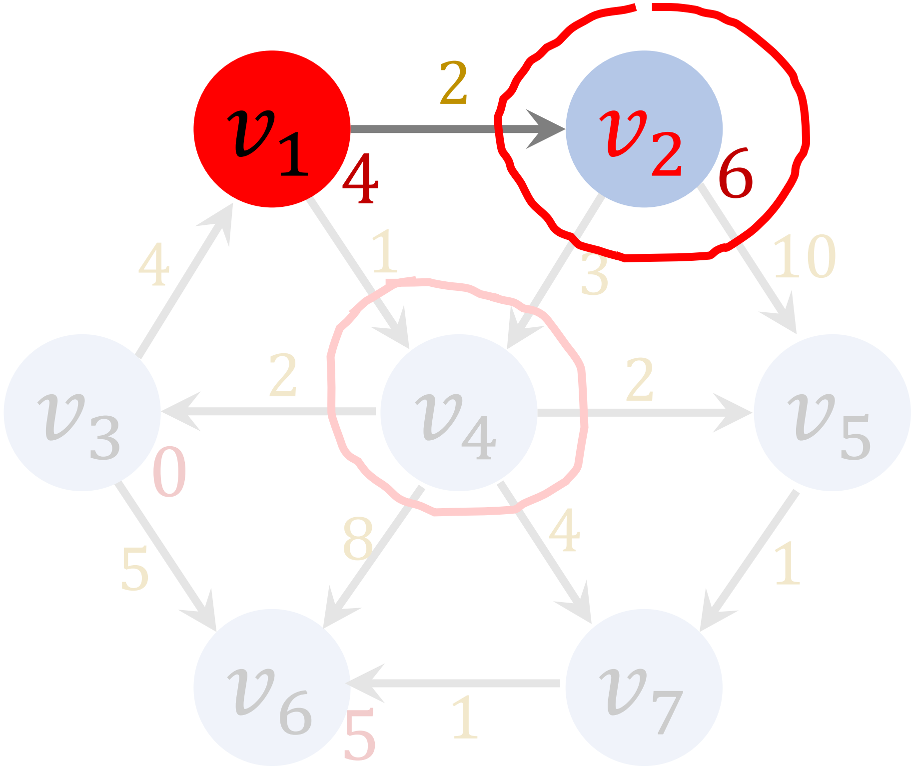
$d_{\text{new}} = 4 + 2 = 6.$

Priority Queue:

vertex	dist
v_6	**5**

vertex	dist	path
v_1	**4**	v_3
v_2	**6**	0
v_3	**0**	0
v_4	∞	0
v_5	∞	0
v_6	**5**	v_3
v_7	∞	0

Iteration 2(A)

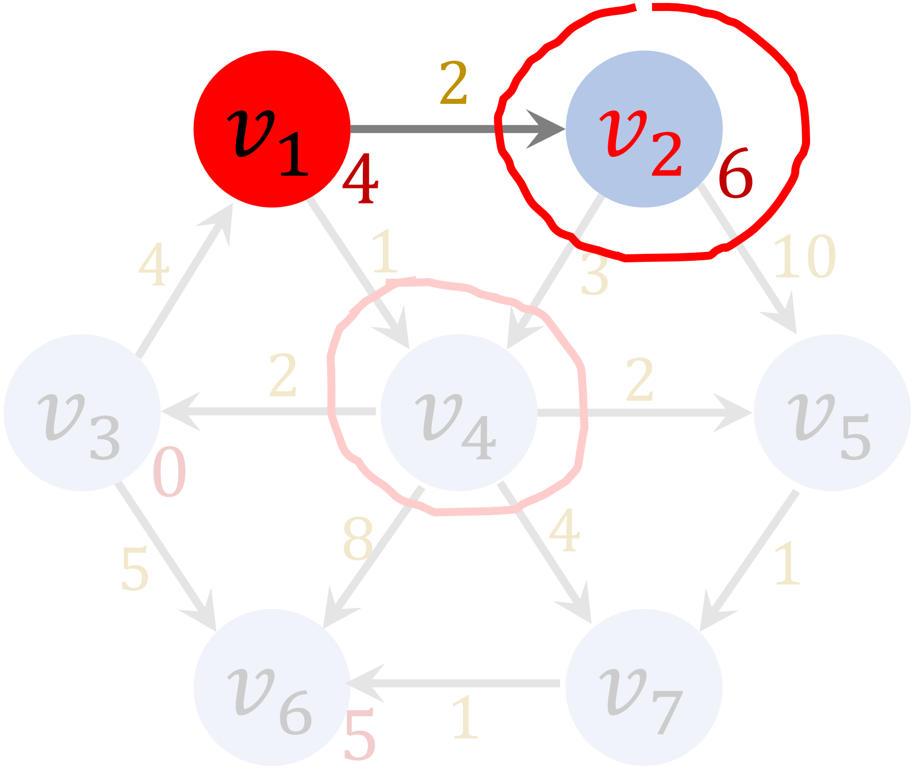


Priority Queue:

vertex	dist
v_6	5

vertex	dist	path
v_1	**4**	v_3
v_2	**6**	v_1
v_3	**0**	0
v_4	∞	0
v_5	∞	0
v_6	**5**	v_3
v_7	∞	0

Iteration 2(A)

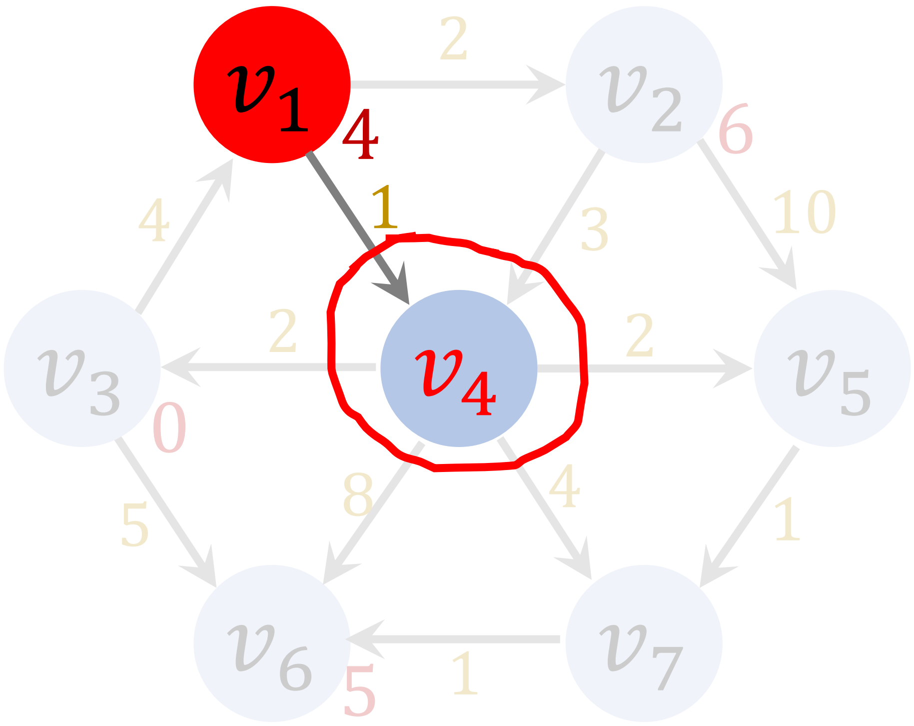


Priority Queue:

vertex	dist
v_6	**5**
v_2	**6**

vertex	dist	path
v_1	**4**	v_3
v_2	**6**	v_1
v_3	**0**	0
v_4	∞	0
v_5	∞	0
v_6	**5**	v_3
v_7	∞	0

Iteration 2(B)

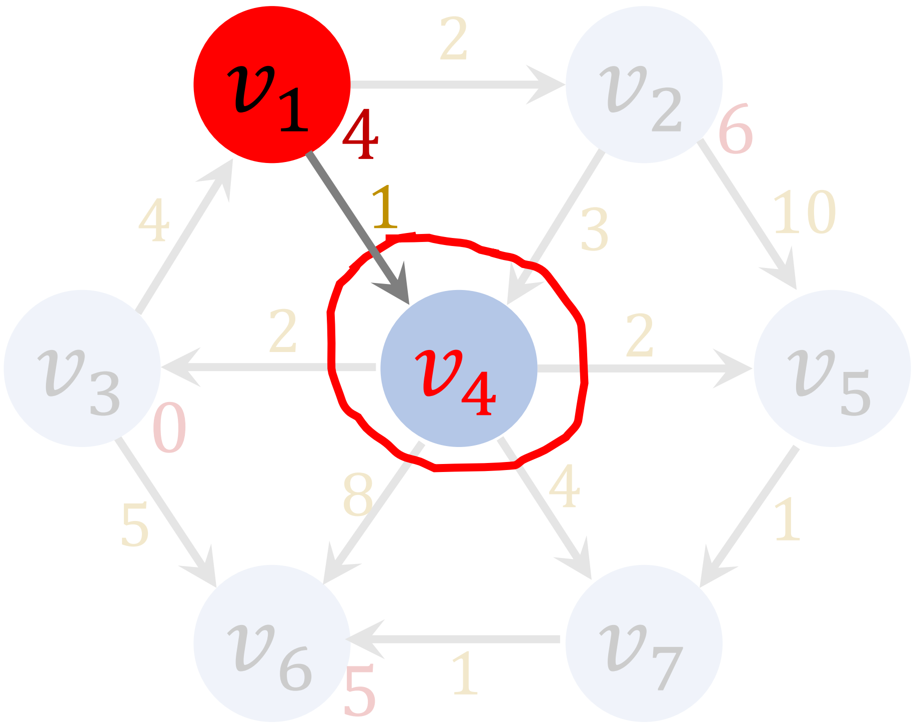


Priority Queue:

vertex	dist
v_6	**5**
v_2	**6**

vertex	dist	path
v_1	**4**	v_3
v_2	**6**	v_1
v_3	**0**	0
v_4	∞	0
v_5	∞	0
v_6	**5**	v_3
v_7	∞	0

Iteration 2(B)



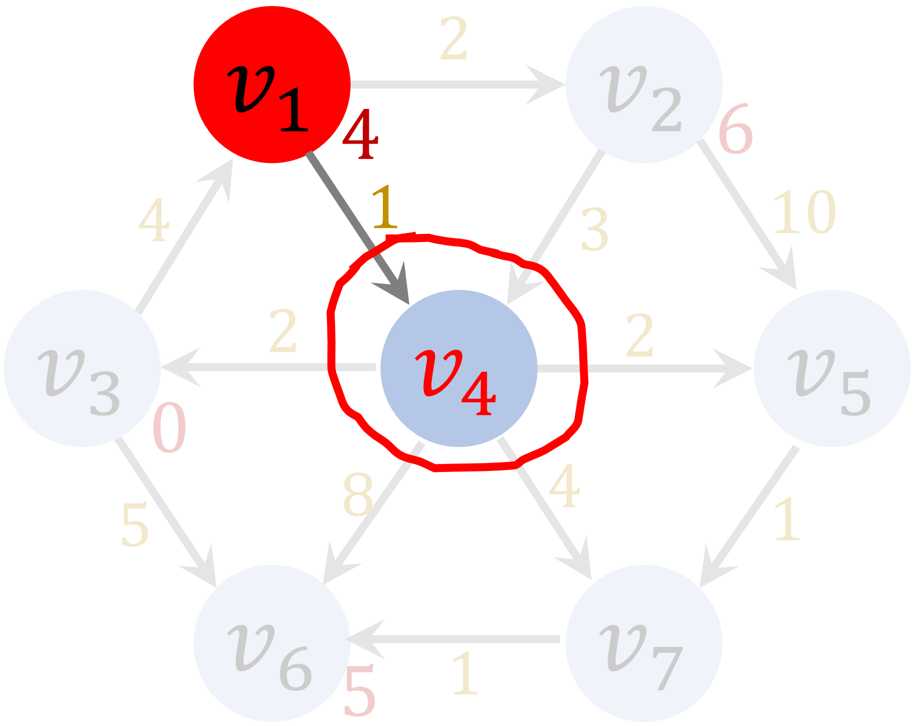
$$d_{\text{new}} = 4 + 1 = 5.$$

Priority Queue:

vertex	dist
v_6	**5**
v_2	**6**

vertex	dist	path
v_1	**4**	v_3
v_2	**6**	v_1
v_3	**0**	0
v_4	∞	0
v_5	∞	0
v_6	**5**	v_3
v_7	∞	0

Iteration 2(B)



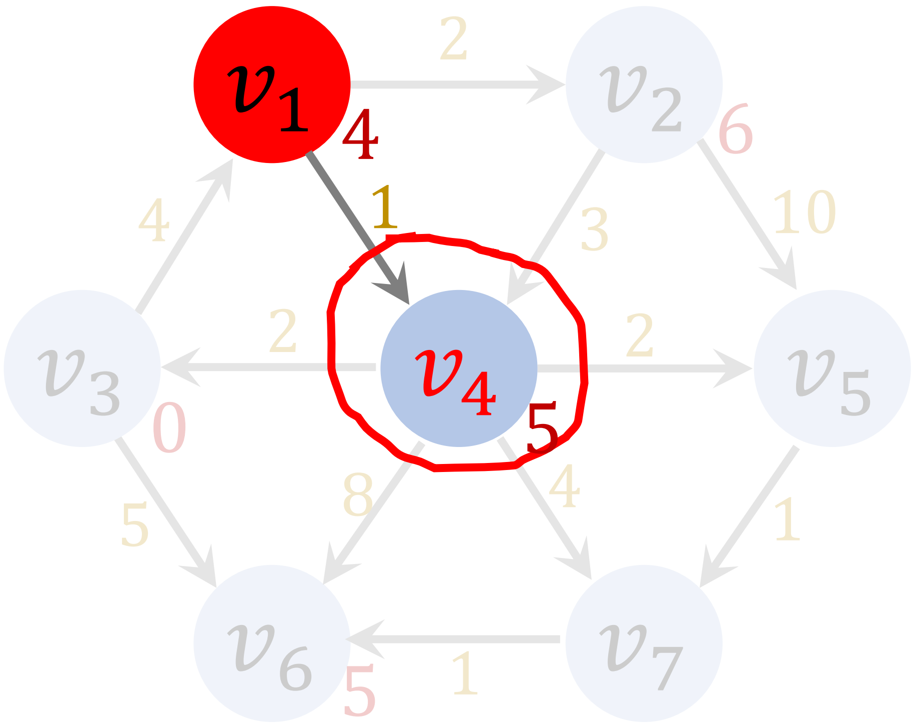
$$d_{\text{new}} = 4 + 1 = 5.$$

Priority Queue:

vertex	dist
v_6	**5**
v_2	**6**

vertex	dist	path
v_1	**4**	v_3
v_2	**6**	v_1
v_3	**0**	0
v_4	∞	0
v_5	∞	0
v_6	**5**	v_3
v_7	∞	0

Iteration 2(B)



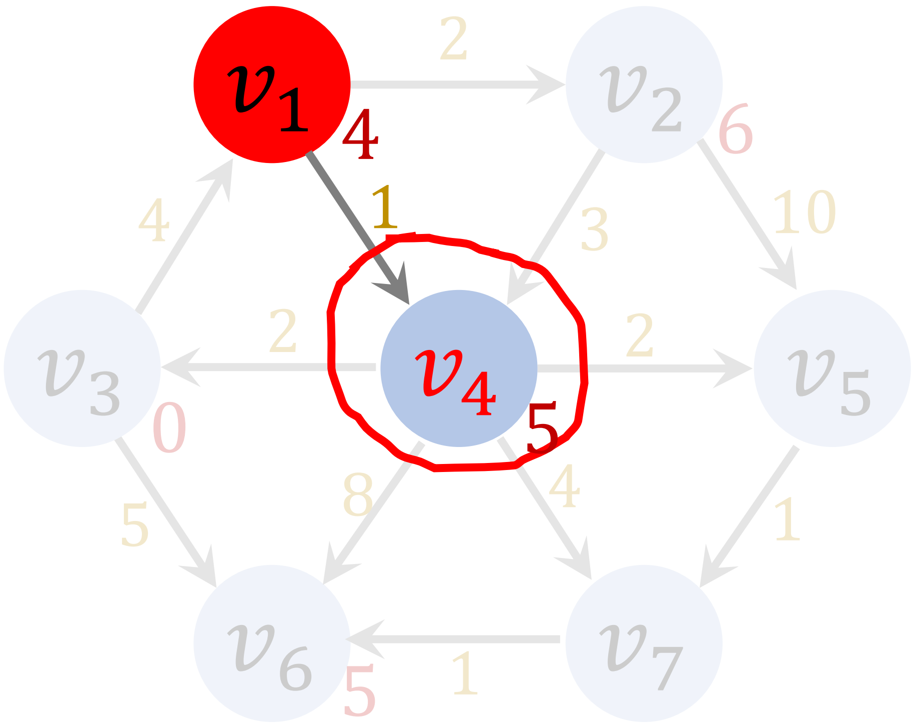
$$d_{\text{new}} = 4 + 1 = 5.$$

Priority Queue:

vertex	dist
v_6	**5**
v_2	**6**

vertex	dist	path
v_1	**4**	v_3
v_2	**6**	v_1
v_3	**0**	0
v_4	**5**	0
v_5	∞	0
v_6	**5**	v_3
v_7	∞	0

Iteration 2(B)

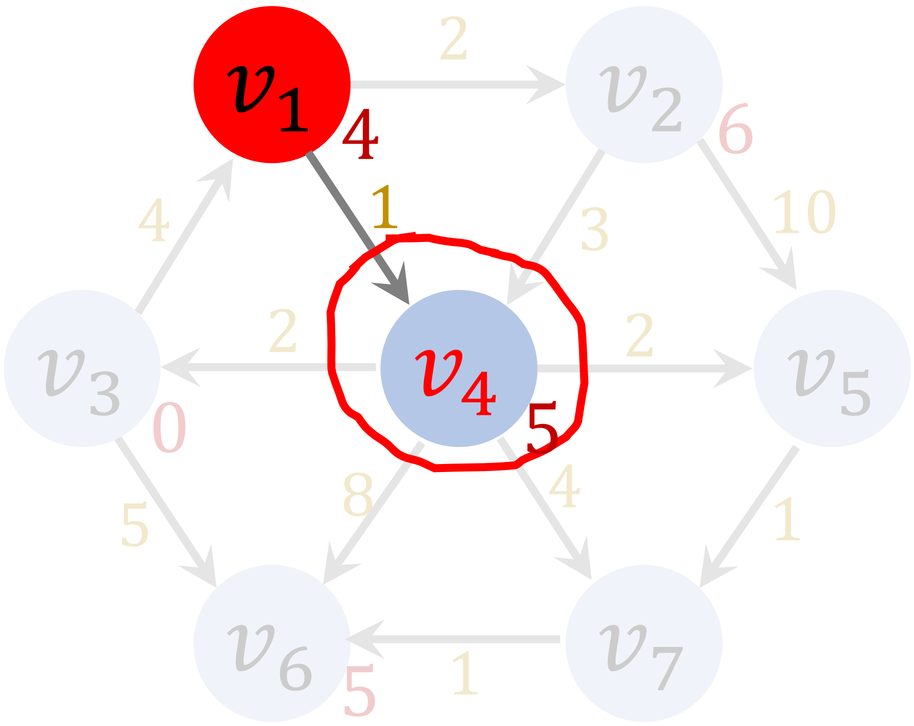


Priority Queue:

vertex	dist
v_6	**5**
v_2	**6**

vertex	dist	path
v_1	**4**	v_3
v_2	**6**	v_1
v_3	**0**	0
v_4	**5**	v_1
v_5	∞	0
v_6	**5**	v_3
v_7	∞	0

Iteration 2(B)

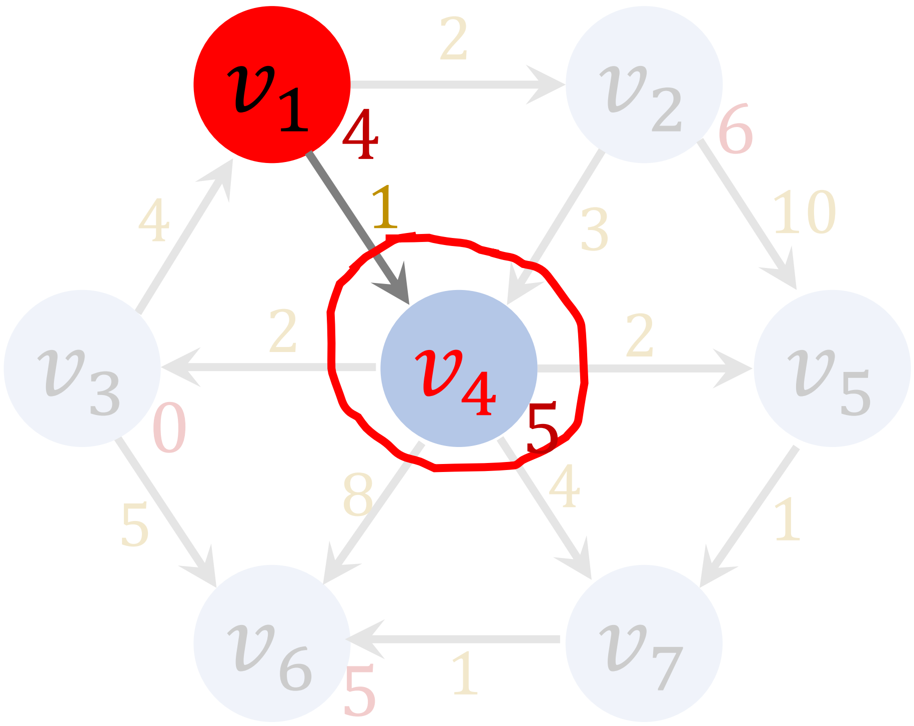


Priority Queue:

vertex	dist
v_6	**5**
v_2	**6**

vertex	dist	path
v_1	**4**	v_3
v_2	**6**	v_1
v_3	**0**	0
v_4	**5**	v_1
v_5	∞	0
v_6	**5**	v_3
v_7	∞	0

Iteration 2(B)

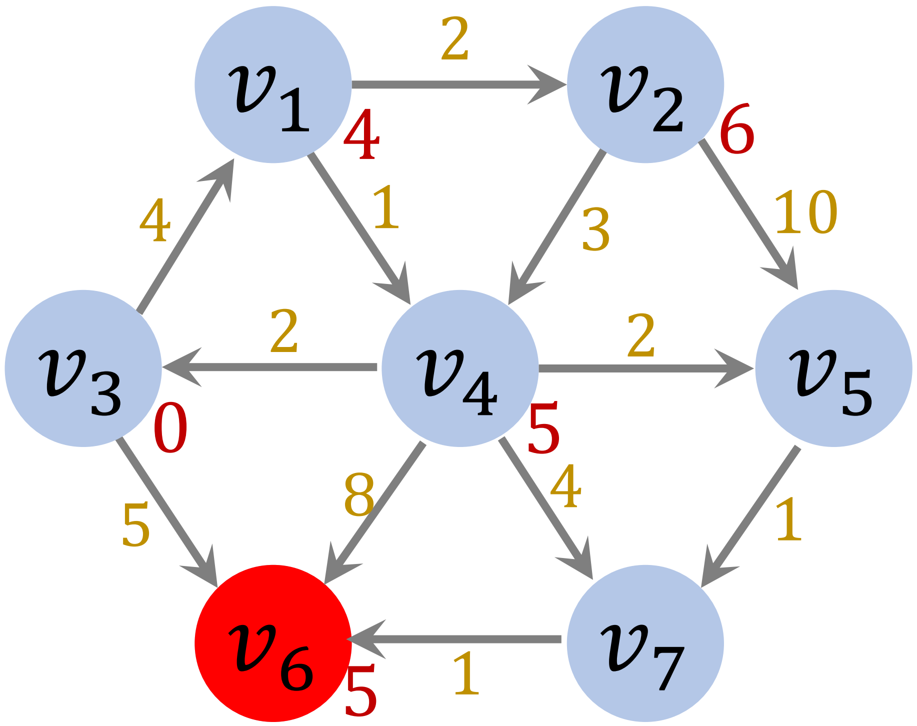


Priority Queue:

vertex	dist
v_6	**5**
v_4	**5**
v_2	**6**

vertex	dist	path
v_1	**4**	v_3
v_2	**6**	v_1
v_3	**0**	0
v_4	**5**	v_1
v_5	∞	0
v_6	**5**	v_3
v_7	∞	0

Iteration 3

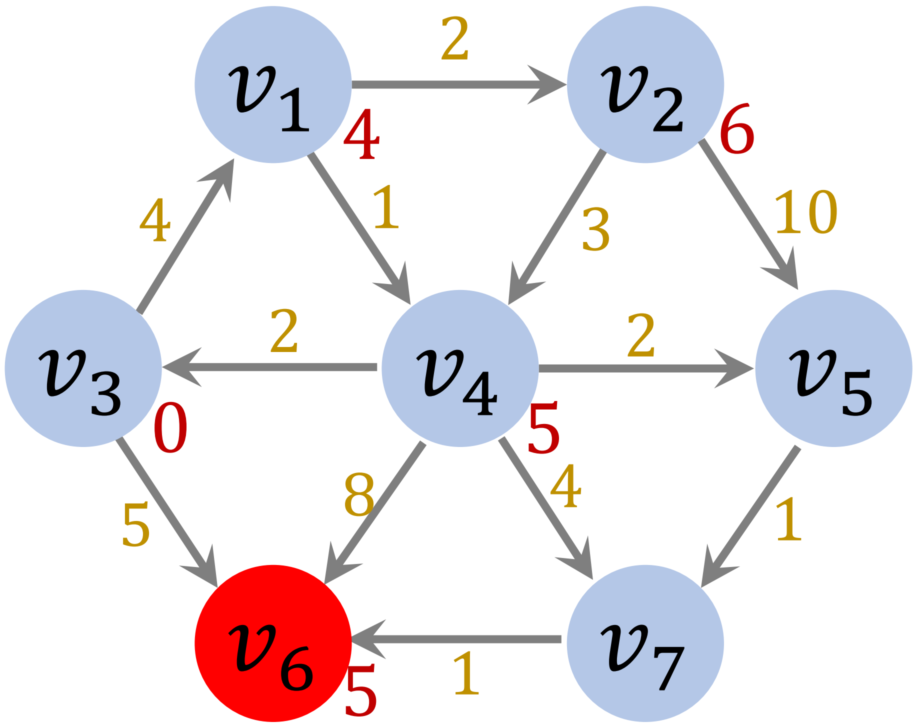


Priority Queue:

vertex	dist
v_6	5
v_4	5
v_2	6

vertex	dist	path
v_1	4	v_3
v_2	6	v_1
v_3	0	0
v_4	5	v_1
v_5	∞	0
v_6	5	v_3
v_7	∞	0

Iteration 3

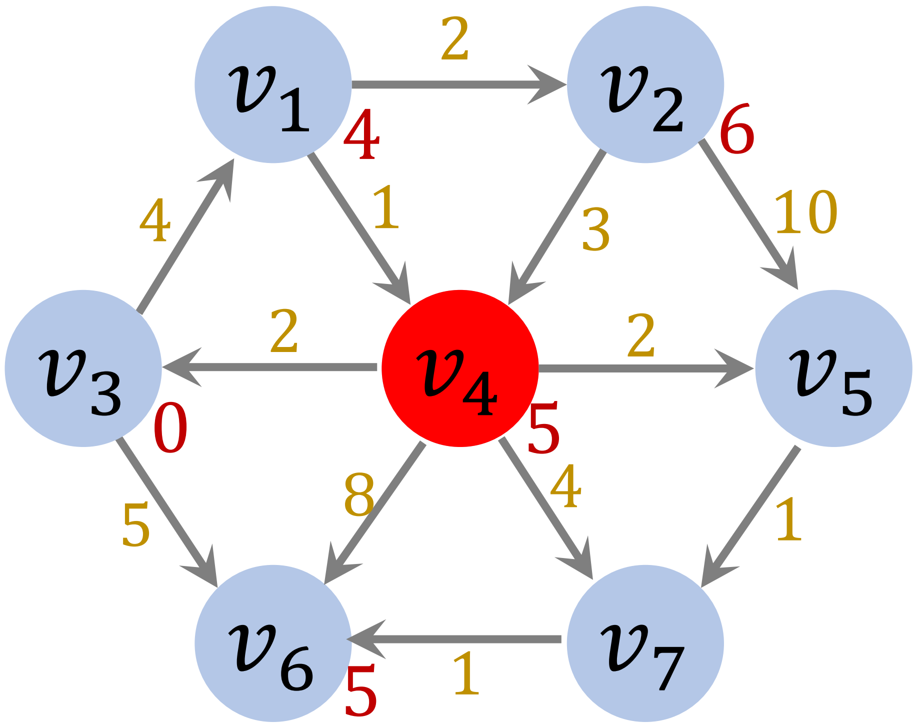


Priority Queue:

vertex	dist
v_4	**5**
v_2	**6**

vertex	dist	path
v_1	**4**	v_3
v_2	**6**	v_1
v_3	**0**	0
v_4	**5**	v_1
v_5	∞	0
v_6	**5**	v_3
v_7	∞	0

Iteration 4

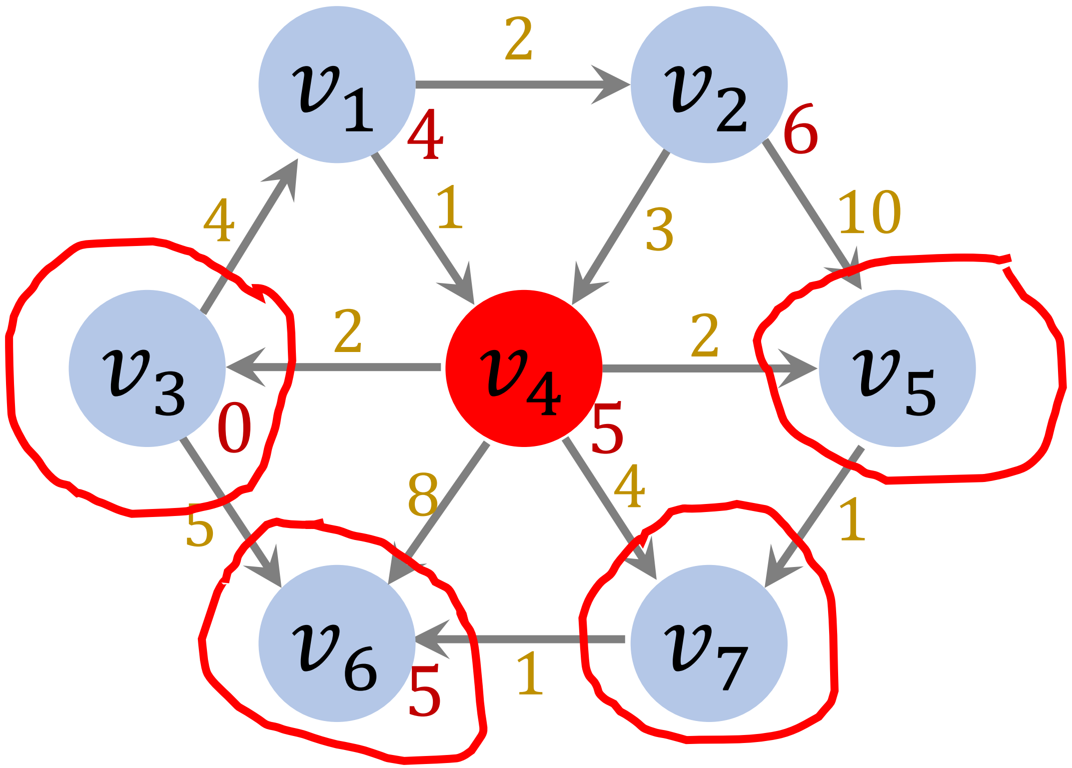


Priority Queue:

vertex	dist
v_4	5
v_2	6

vertex	dist	path
v_1	4	v_3
v_2	6	v_1
v_3	0	0
v_4	5	v_1
v_5	∞	0
v_6	5	v_3
v_7	∞	0

Iteration 4

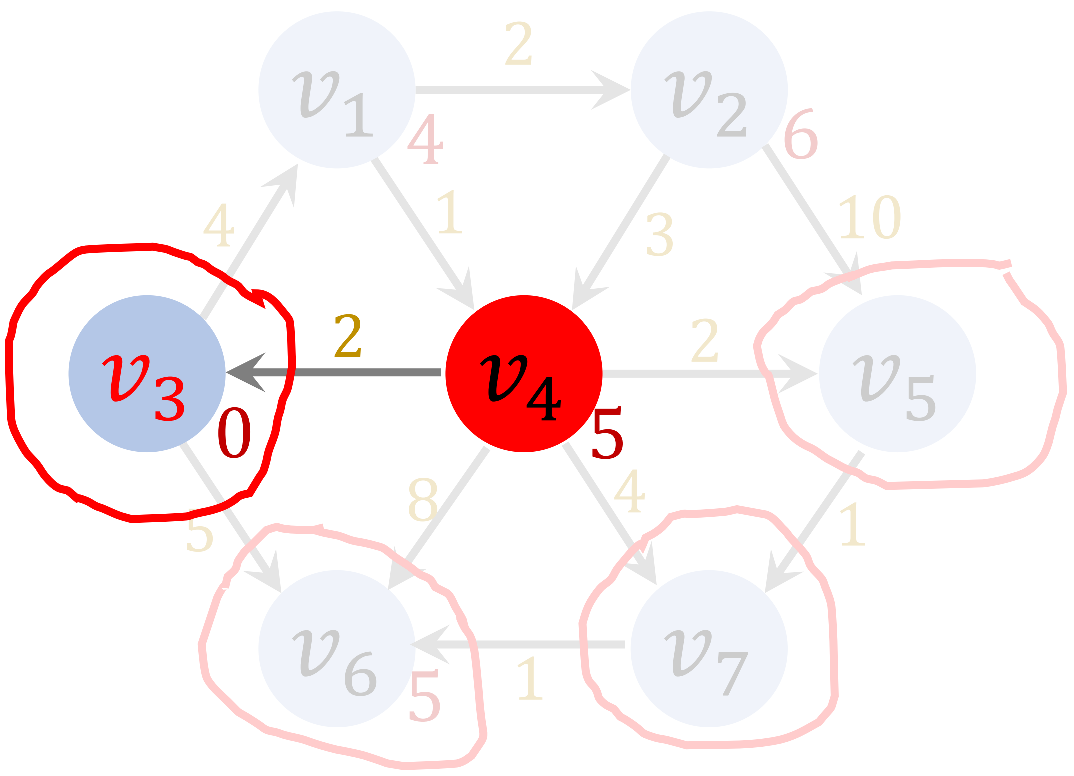


Priority Queue:

vertex	dist
v_2	**6**

vertex	dist	path
v_1	**4**	v_3
v_2	**6**	v_1
v_3	**0**	0
v_4	**5**	v_1
v_5	∞	0
v_6	**5**	v_3
v_7	∞	0

Iteration 4(A)

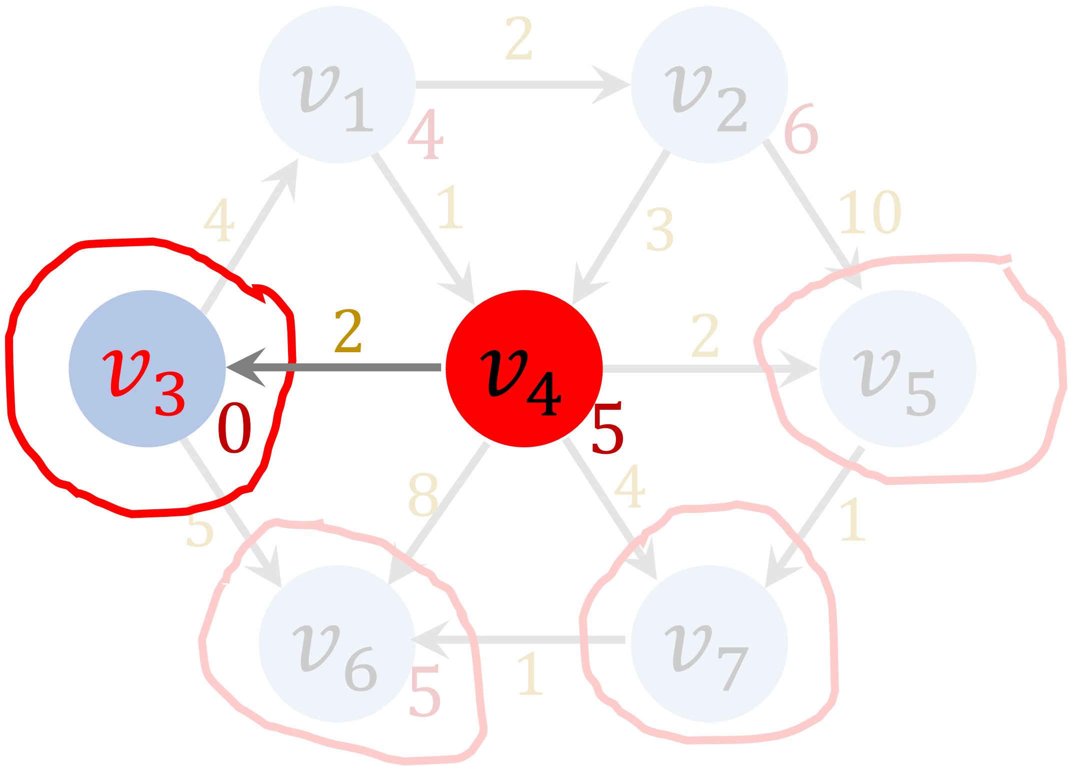


Priority Queue:

vertex	dist
v_2	**6**

vertex	dist	path
v_1	**4**	v_3
v_2	**6**	v_1
v_3	**0**	0
v_4	**5**	v_1
v_5	∞	0
v_6	**5**	v_3
v_7	∞	0

Iteration 4(A)



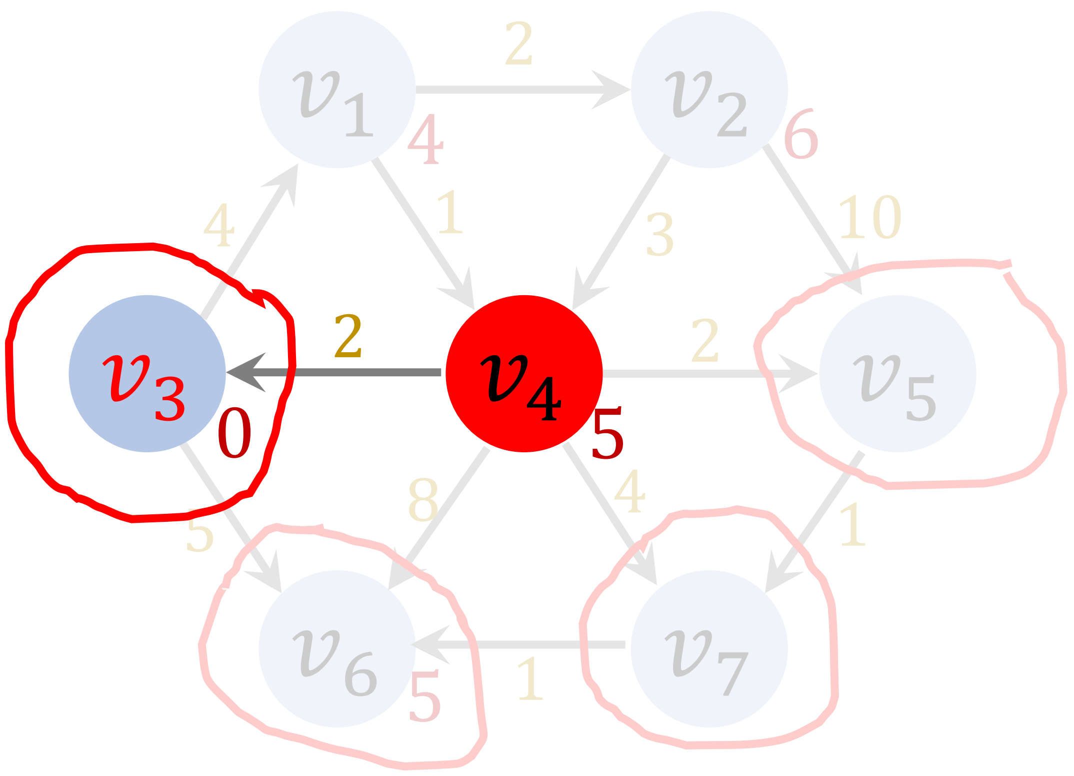
$$d_{\text{new}} = 5 + 2 = 7.$$

Priority Queue:

vertex	dist
v_2	6

vertex	dist	path
v_1	**4**	v_3
v_2	**6**	v_1
v_3	**0**	0
v_4	**5**	v_1
v_5	∞	0
v_6	**5**	v_3
v_7	∞	0

Iteration 4(A)



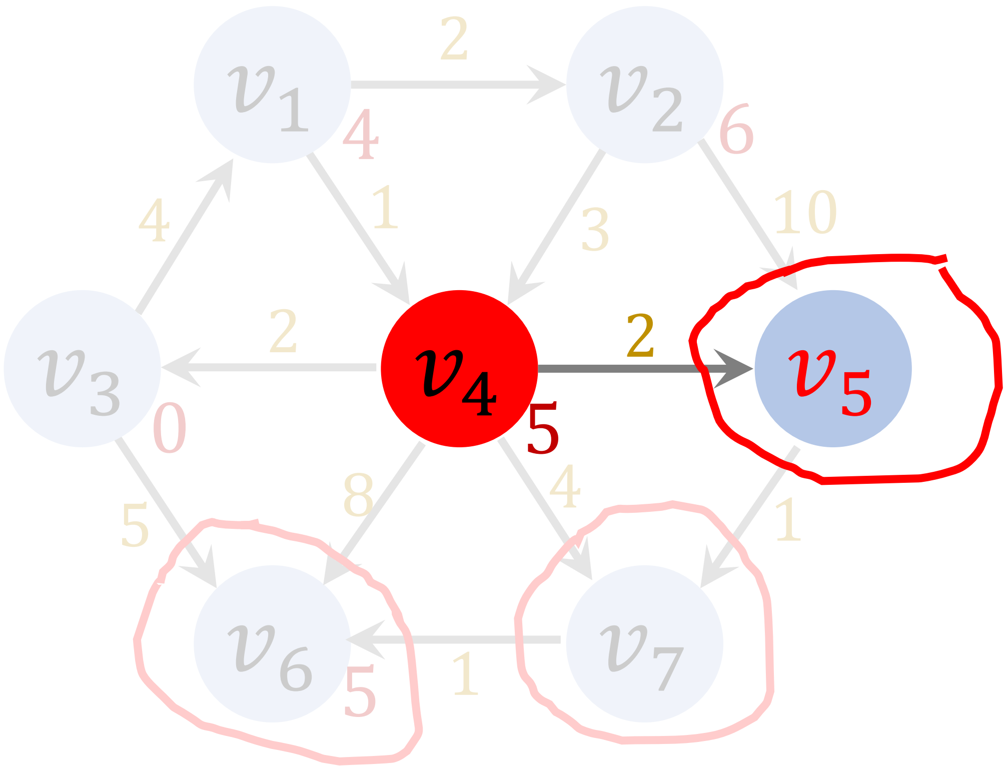
$d_{\text{new}} = 5 + 2 = 7.$

Priority Queue:

vertex	dist
v_2	6

vertex	dist	path
v_1	4	v_3
v_2	6	v_1
v_3	0	0
v_4	5	v_1
v_5	∞	0
v_6	5	v_3
v_7	∞	0

Iteration 4(B)

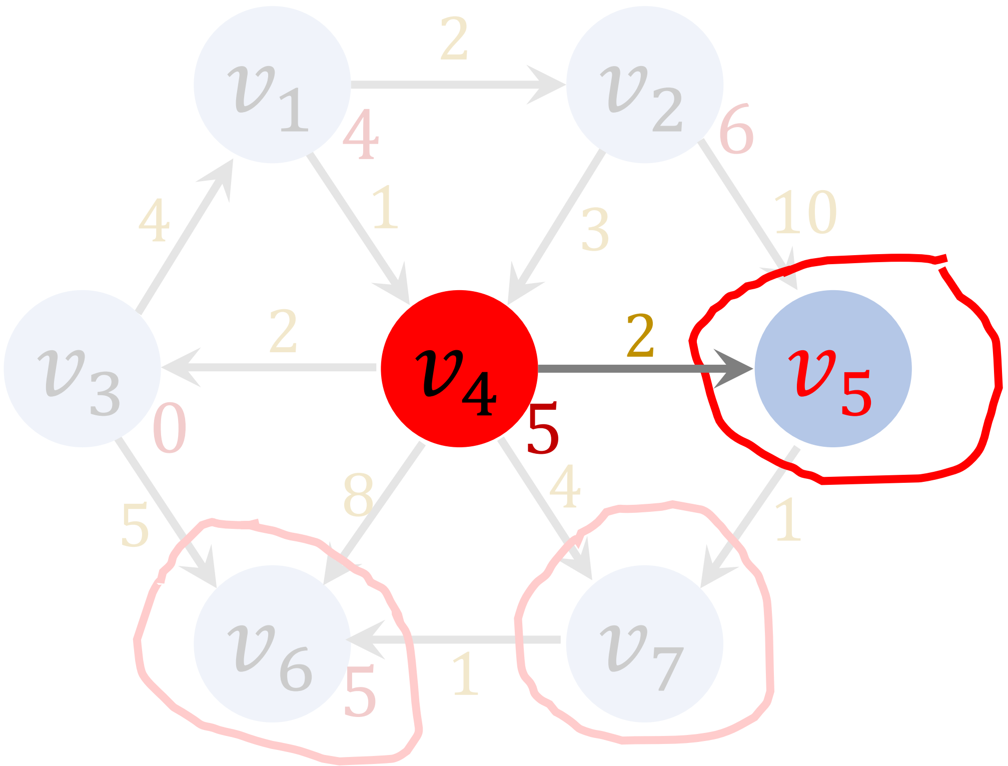


Priority Queue:

vertex	dist
v_2	6

vertex	dist	path
v_1	**4**	v_3
v_2	**6**	v_1
v_3	**0**	0
v_4	**5**	v_1
v_5	∞	0
v_6	**5**	v_3
v_7	∞	0

Iteration 4(B)



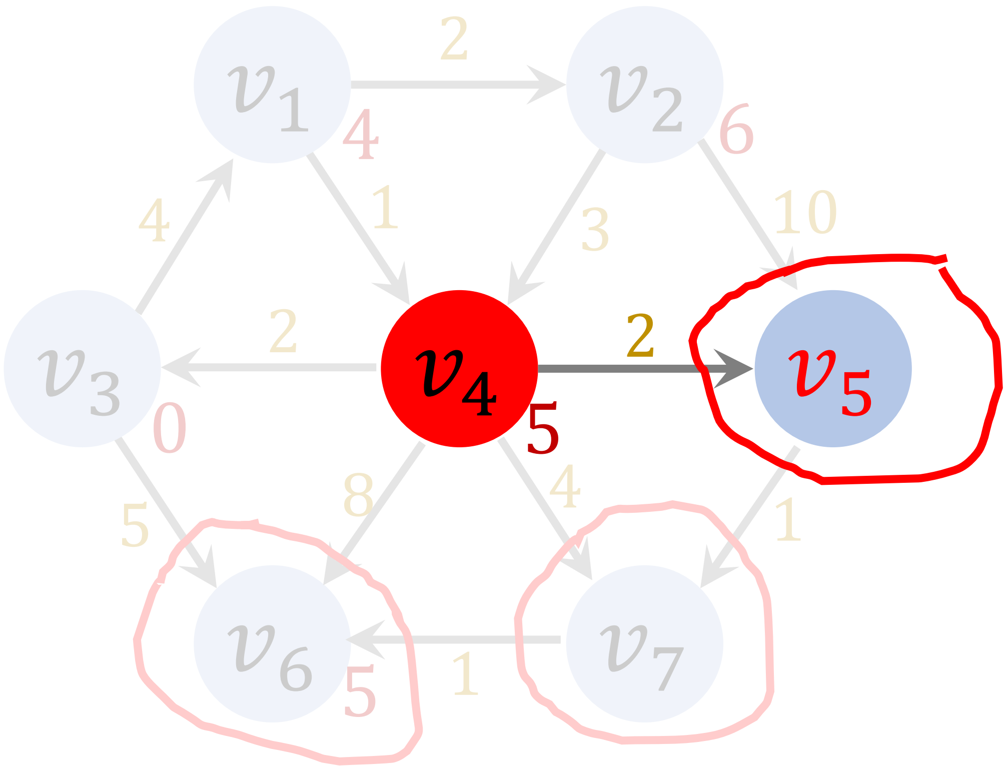
$$d_{\text{new}} = 5 + 2 = 7.$$

Priority Queue:

vertex	dist
v_2	6

vertex	dist	path
v_1	**4**	v_3
v_2	**6**	v_1
v_3	**0**	0
v_4	**5**	v_1
v_5	∞	0
v_6	**5**	v_3
v_7	∞	0

Iteration 4(B)



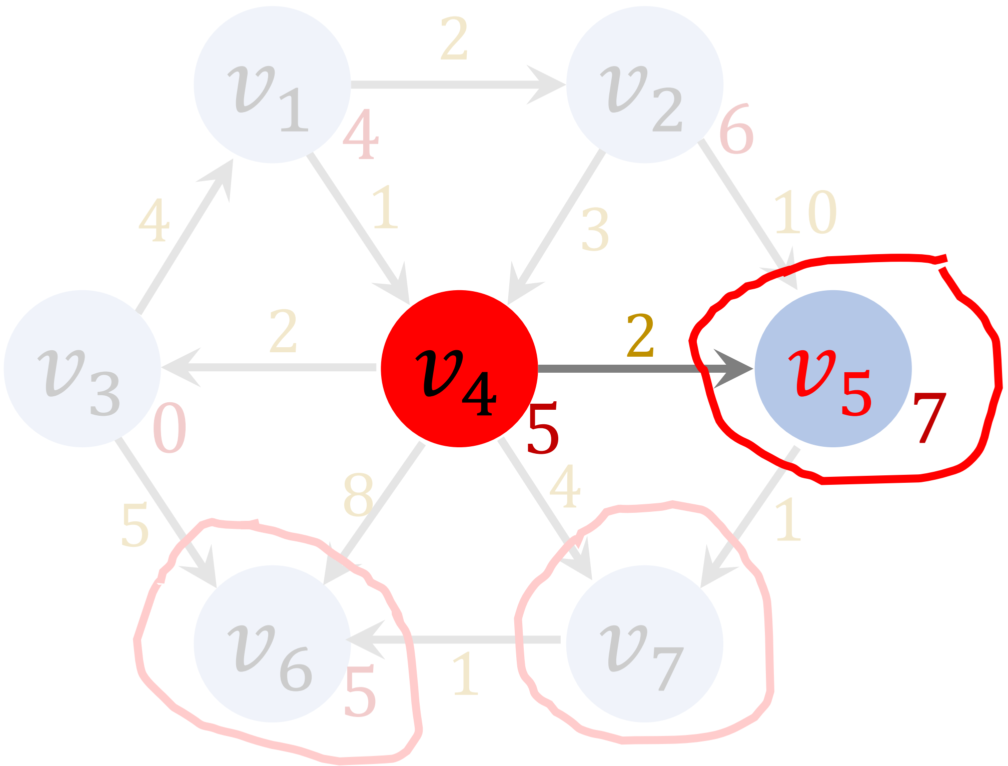
$$d_{\text{new}} = 5 + 2 = 7.$$

Priority Queue:

vertex	dist
v_2	6

vertex	dist	path
v_1	**4**	v_3
v_2	**6**	v_1
v_3	**0**	0
v_4	**5**	v_1
v_5	∞	0
v_6	**5**	v_3
v_7	∞	0

Iteration 4(B)



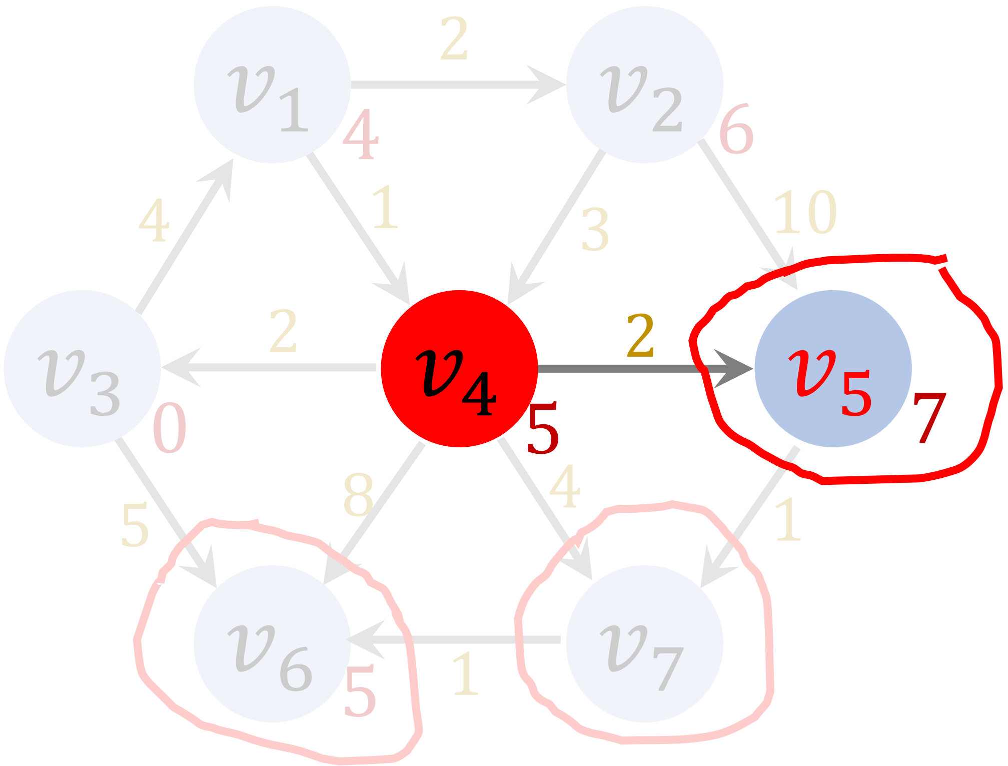
$$d_{\text{new}} = 5 + 2 = 7.$$

Priority Queue:

vertex	dist
v_2	6

vertex	dist	path
v_1	4	v_3
v_2	6	v_1
v_3	0	0
v_4	5	v_1
v_5	7	0
v_6	5	v_3
v_7	∞	0

Iteration 4(B)

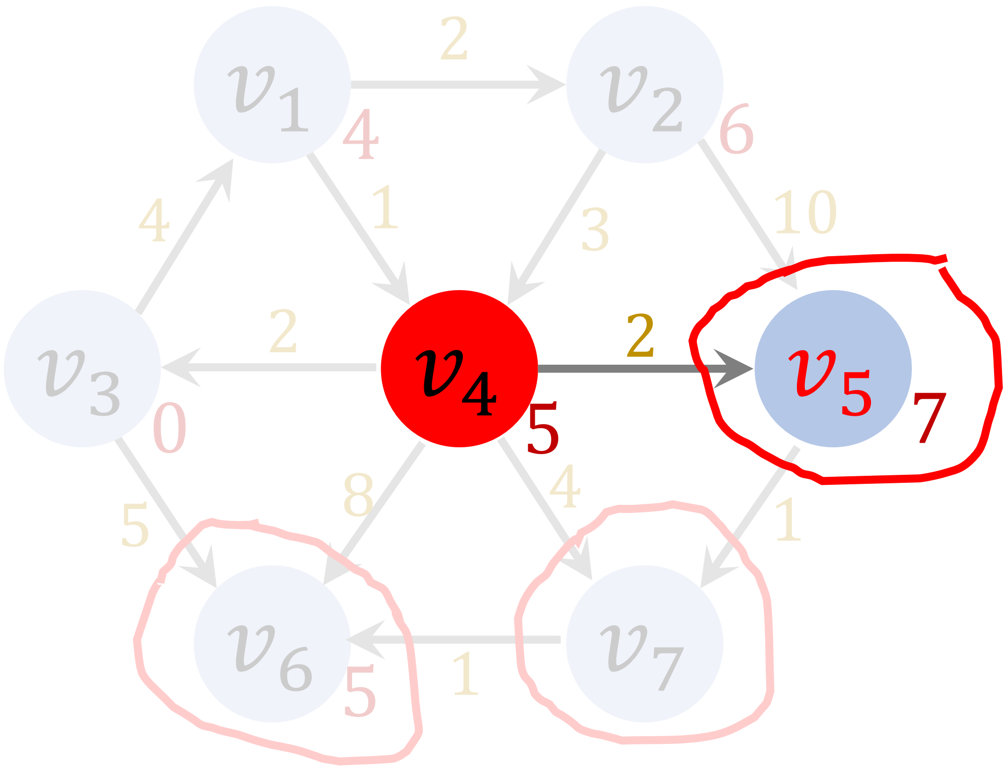


Priority Queue:

vertex	dist
v_2	6

vertex	dist	path
v_1	4	v_3
v_2	6	v_1
v_3	0	0
v_4	5	v_1
v_5	7	v_4
v_6	5	v_3
v_7	∞	0

Iteration 4(B)

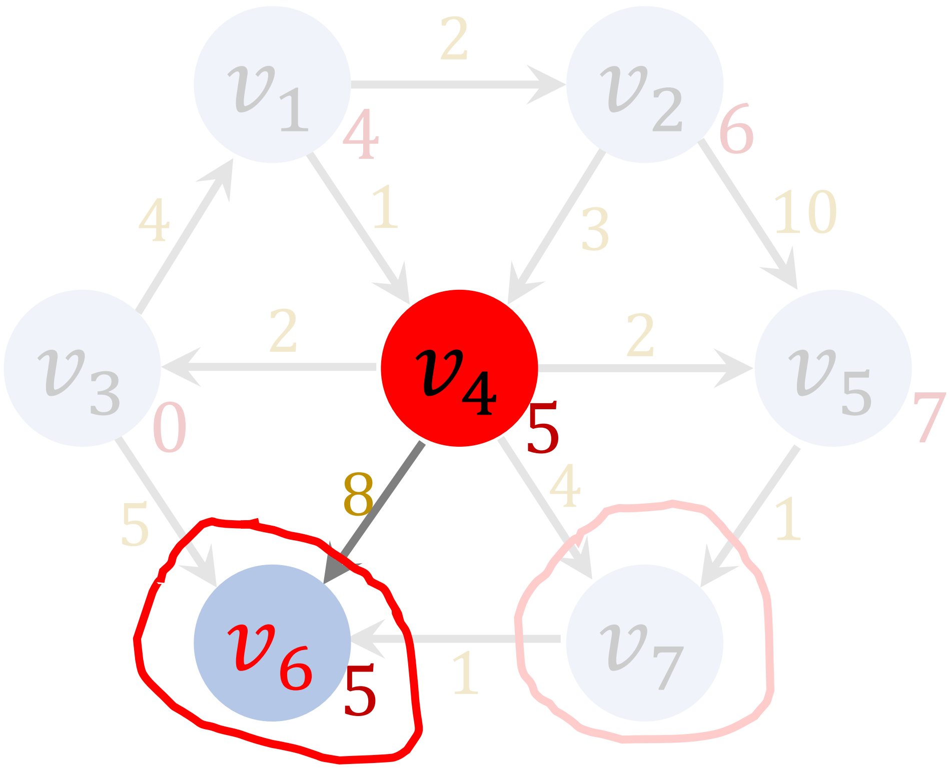


Priority Queue:

vertex	dist
v_2	**6**
v_5	**7**

vertex	dist	path
v_1	**4**	v_3
v_2	**6**	v_1
v_3	**0**	0
v_4	**5**	v_1
v_5	**7**	v_4
v_6	**5**	v_3
v_7	∞	0

Iteration 4(C)

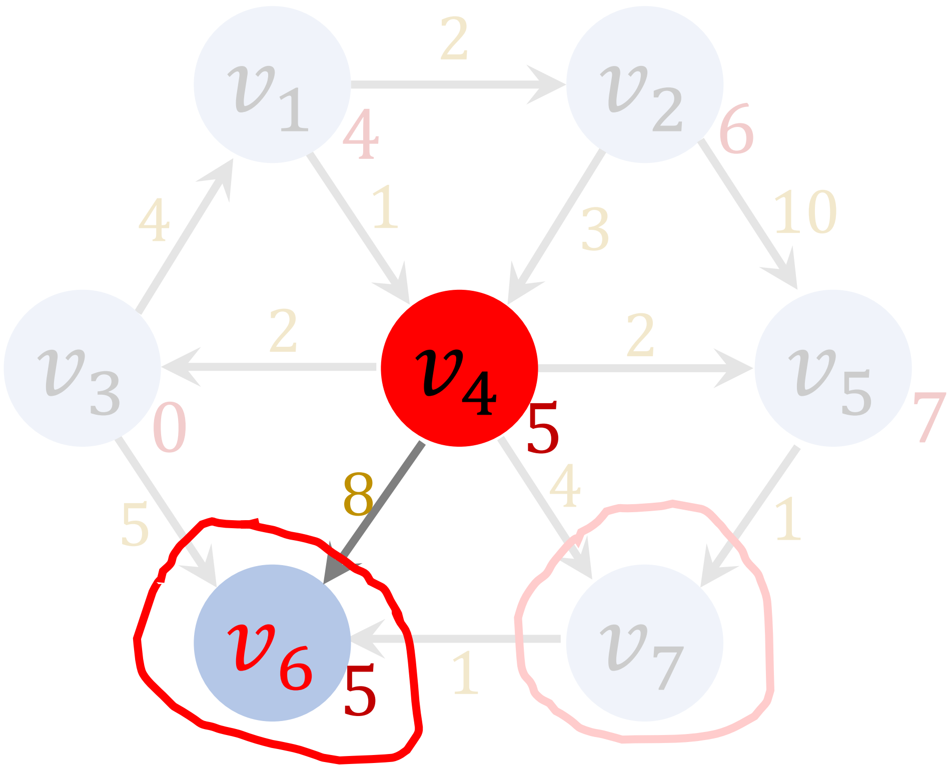


Priority Queue:

vertex	dist
v_2	**6**
v_5	**7**

vertex	dist	path
v_1	**4**	v_3
v_2	**6**	v_1
v_3	**0**	0
v_4	**5**	v_1
v_5	**7**	v_4
v_6	**5**	v_3
v_7	∞	0

Iteration 4(C)



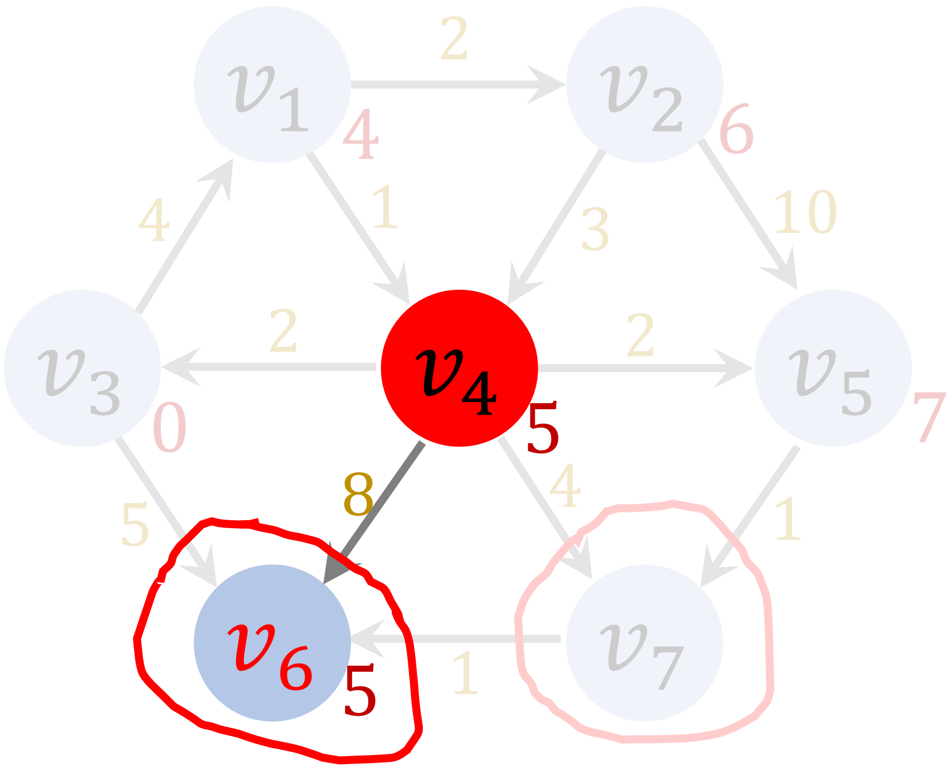
$$d_{\text{new}} = 5 + 8 = 13.$$

Priority Queue:

vertex	dist
v_2	6
v_5	7

vertex	dist	path
v_1	**4**	v_3
v_2	**6**	v_1
v_3	**0**	0
v_4	**5**	v_1
v_5	**7**	v_4
v_6	**5**	v_3
v_7	∞	0

Iteration 4(C)



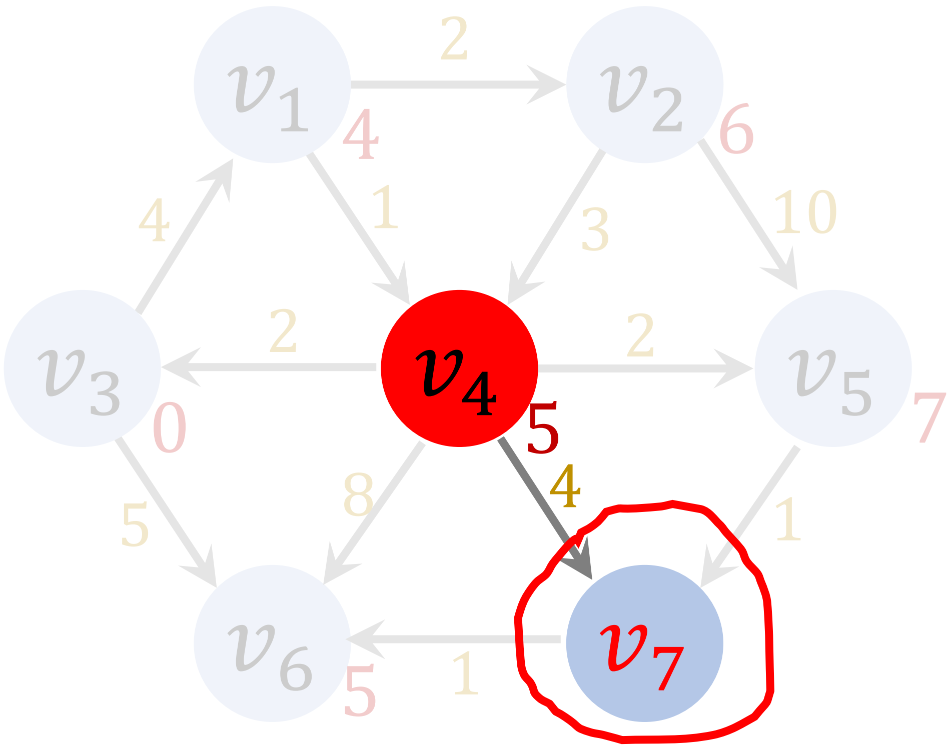
$$d_{\text{new}} = 5 + 8 = 13.$$

Priority Queue:

vertex	dist
v_2	6
v_5	7

vertex	dist	path
v_1	4	v_3
v_2	6	v_1
v_3	0	0
v_4	5	v_1
v_5	7	v_4
v_6	5	v_3
v_7	∞	0

Iteration 4(D)

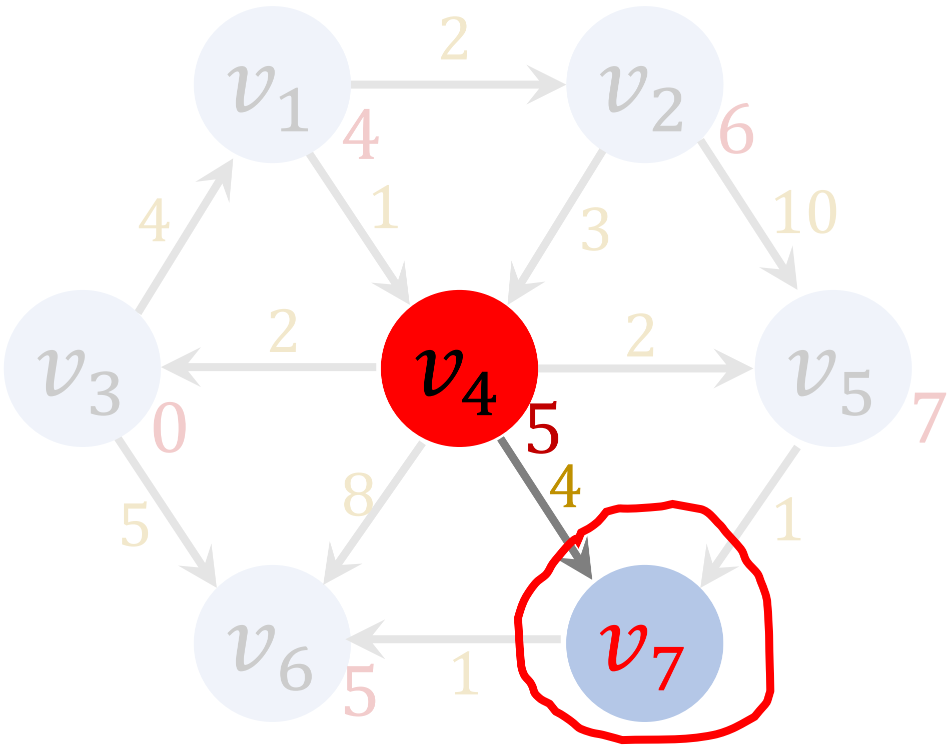


Priority Queue:

vertex	dist
v_2	**6**
v_5	**7**

vertex	dist	path
v_1	**4**	v_3
v_2	**6**	v_1
v_3	**0**	0
v_4	**5**	v_1
v_5	**7**	v_4
v_6	**5**	v_3
v_7	∞	0

Iteration 4(D)



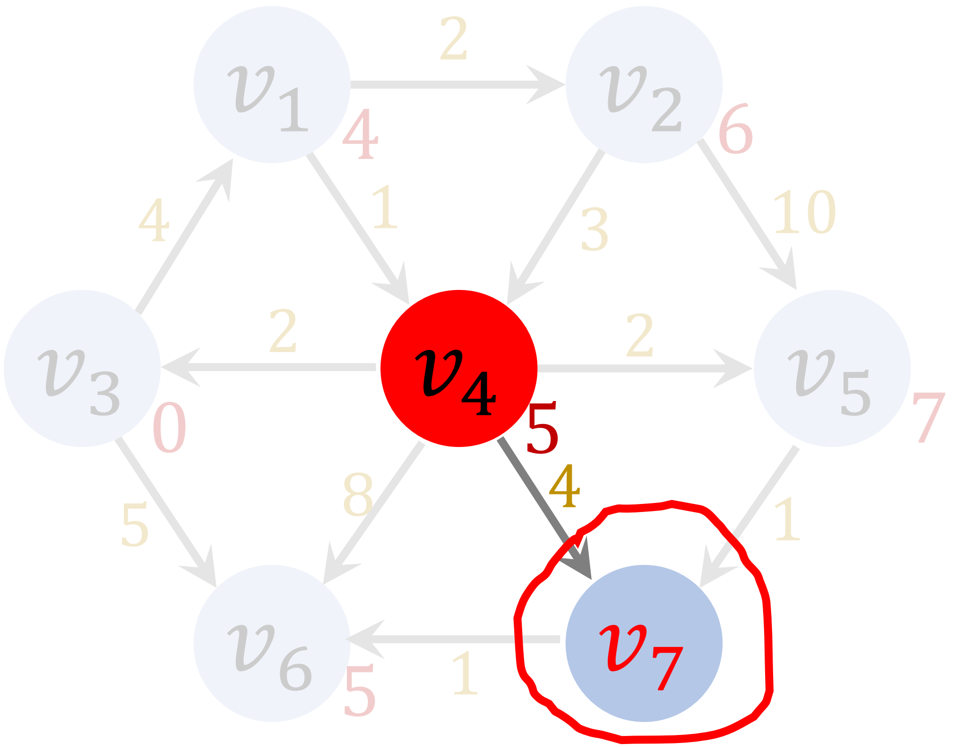
$$d_{\text{new}} = 5 + 4 = 9.$$

Priority Queue:

vertex	dist
v_2	6
v_5	7

vertex	dist	path
v_1	4	v_3
v_2	6	v_1
v_3	0	0
v_4	5	v_1
v_5	7	v_4
v_6	5	v_3
v_7	∞	0

Iteration 4(D)



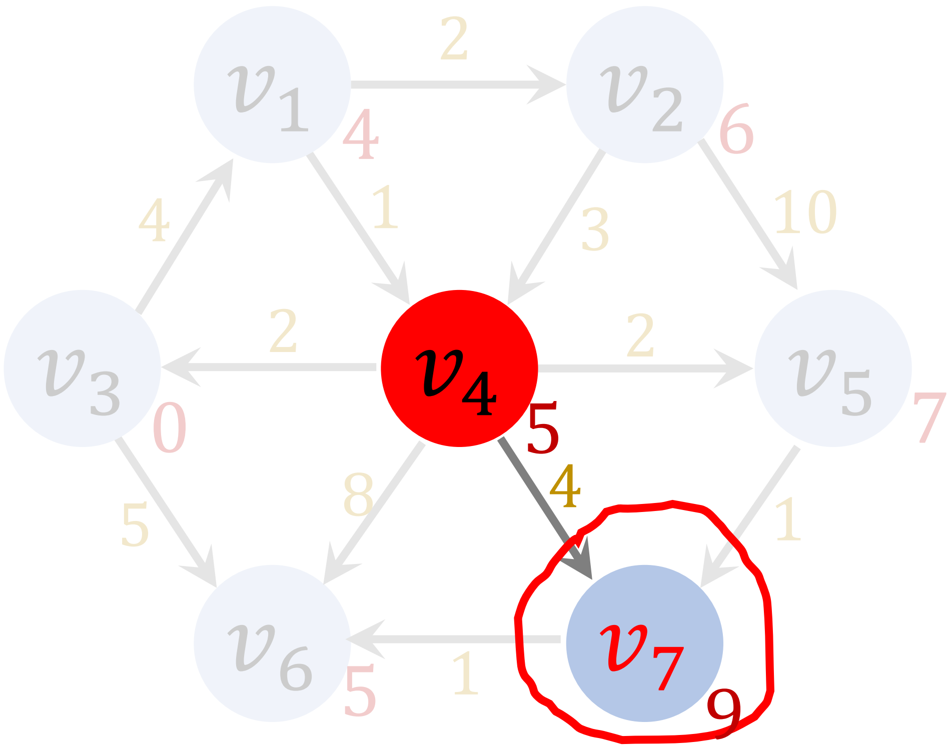
$$d_{\text{new}} = 5 + 4 = 9.$$

Priority Queue:

vertex	dist
v_2	6
v_5	7

vertex	dist	path
v_1	4	v_3
v_2	6	v_1
v_3	0	0
v_4	5	v_1
v_5	7	v_4
v_6	5	v_3
v_7	∞	0

Iteration 4(D)



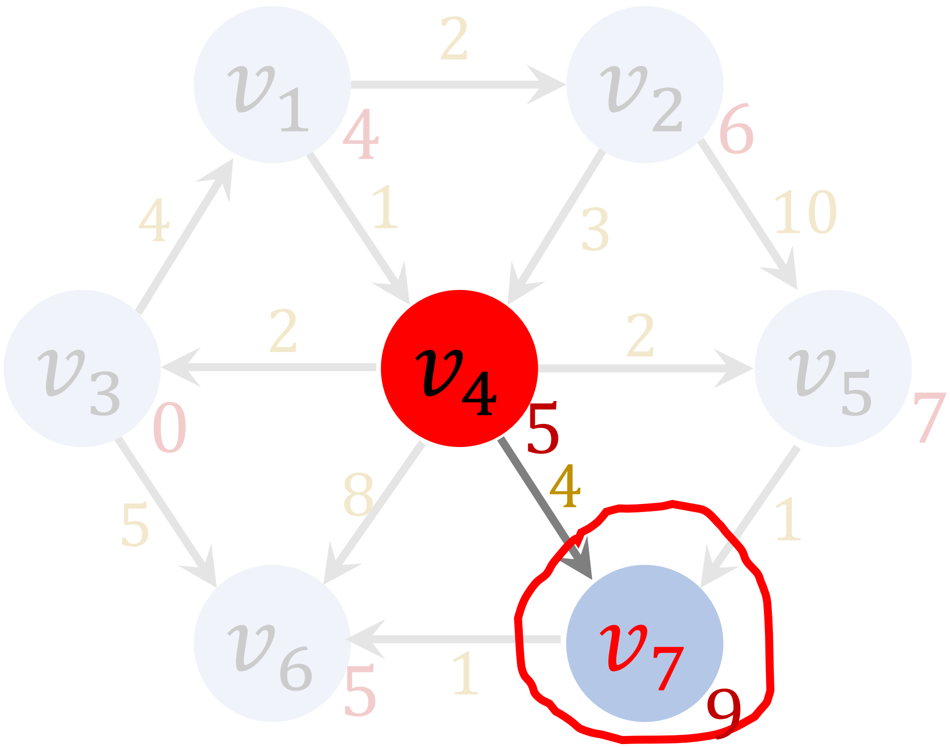
$$d_{\text{new}} = 5 + 4 = 9.$$

Priority Queue:

vertex	dist
v_2	**6**
v_5	**7**

vertex	dist	path
v_1	**4**	v_3
v_2	**6**	v_1
v_3	**0**	0
v_4	**5**	v_1
v_5	**7**	v_4
v_6	**5**	v_3
v_7	**9**	0

Iteration 4(D)

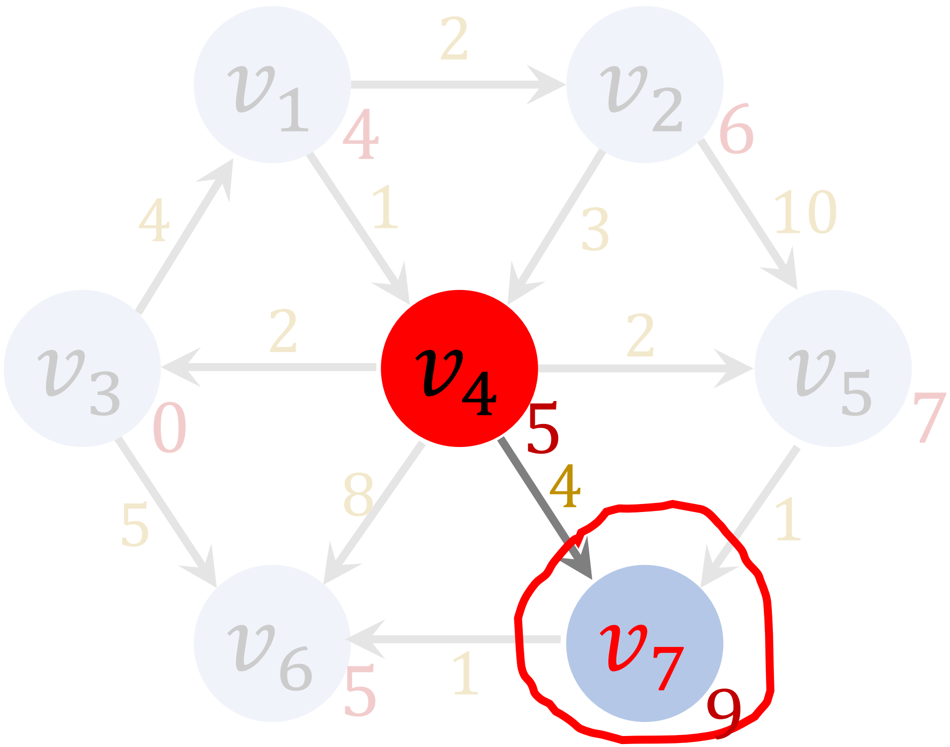


Priority Queue:

vertex	dist
v_2	6
v_5	7

vertex	dist	path
v_1	4	v_3
v_2	6	v_1
v_3	0	0
v_4	5	v_1
v_5	7	v_4
v_6	5	v_3
v_7	9	v_4

Iteration 4(D)

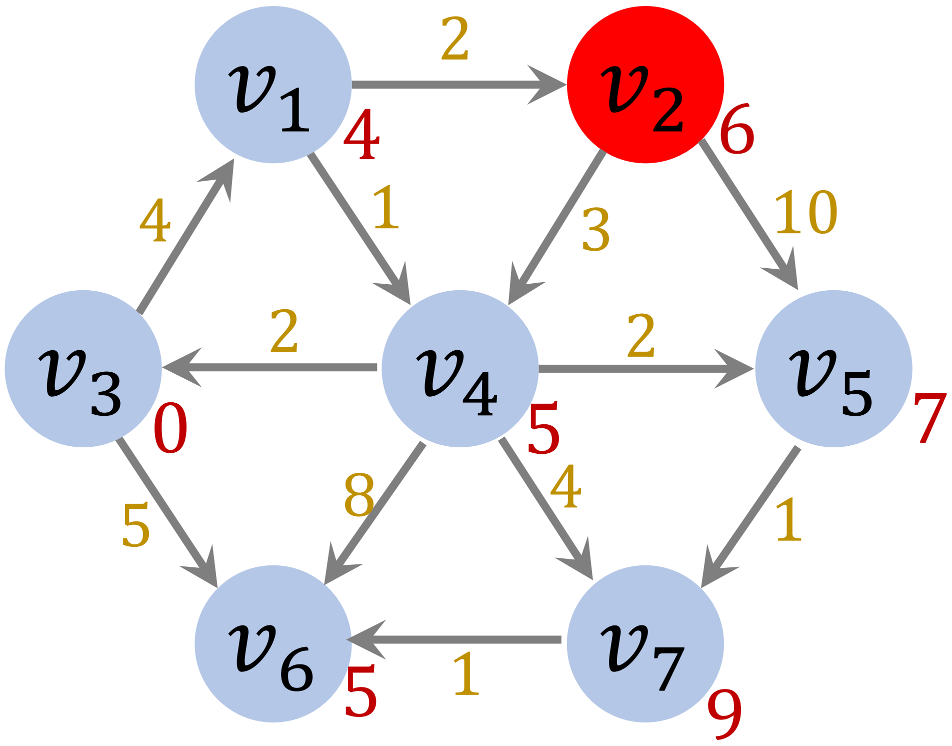


Priority Queue:

vertex	dist
v_2	**6**
v_5	**7**
v_7	**9**

vertex	dist	path
v_1	**4**	v_3
v_2	**6**	v_1
v_3	**0**	0
v_4	**5**	v_1
v_5	**7**	v_4
v_6	**5**	v_3
v_7	**9**	v_4

Iteration 5

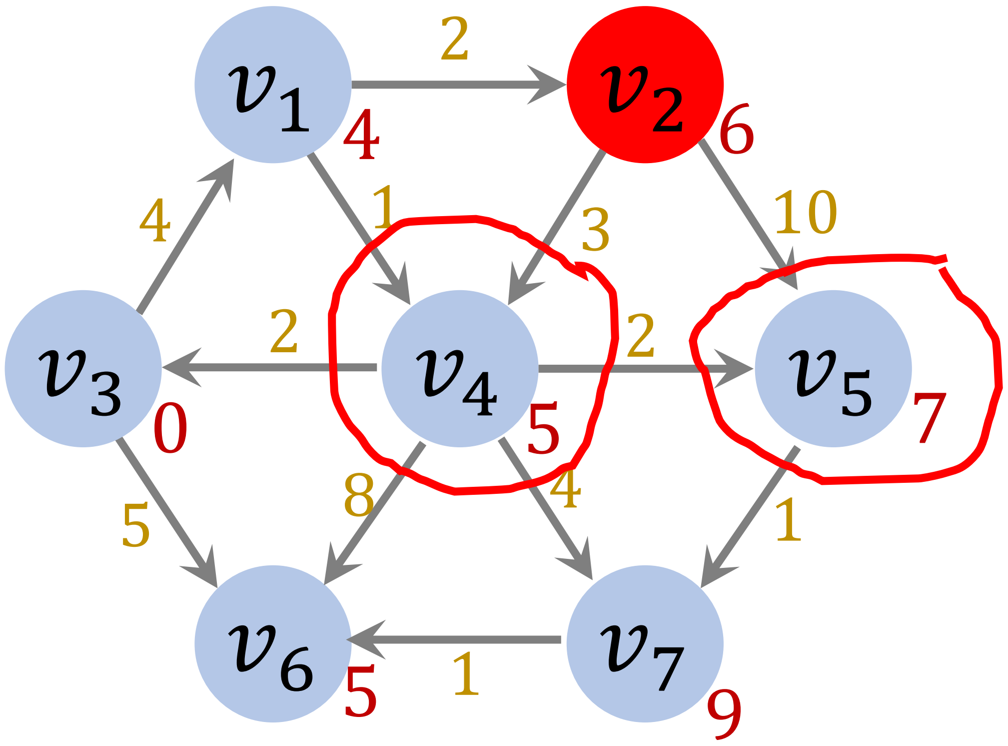


Priority Queue:

vertex	dist
v_2	6
v_5	7
v_7	9

vertex	dist	path
v_1	**4**	v_3
v_2	**6**	v_1
v_3	**0**	0
v_4	**5**	v_1
v_5	**7**	v_4
v_6	**5**	v_3
v_7	**9**	v_4

Iteration 5

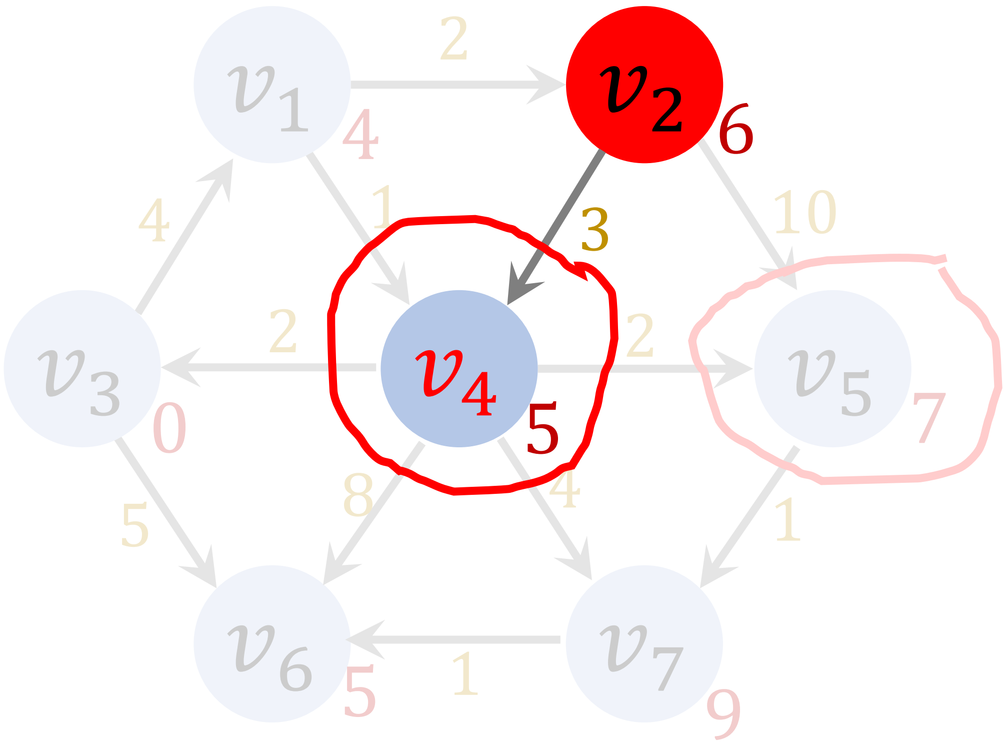


Priority Queue:

vertex	dist
v_5	7
v_7	9

vertex	dist	path
v_1	4	v_3
v_2	6	v_1
v_3	0	0
v_4	5	v_1
v_5	7	v_4
v_6	5	v_3
v_7	9	v_4

Iteration 5(A)

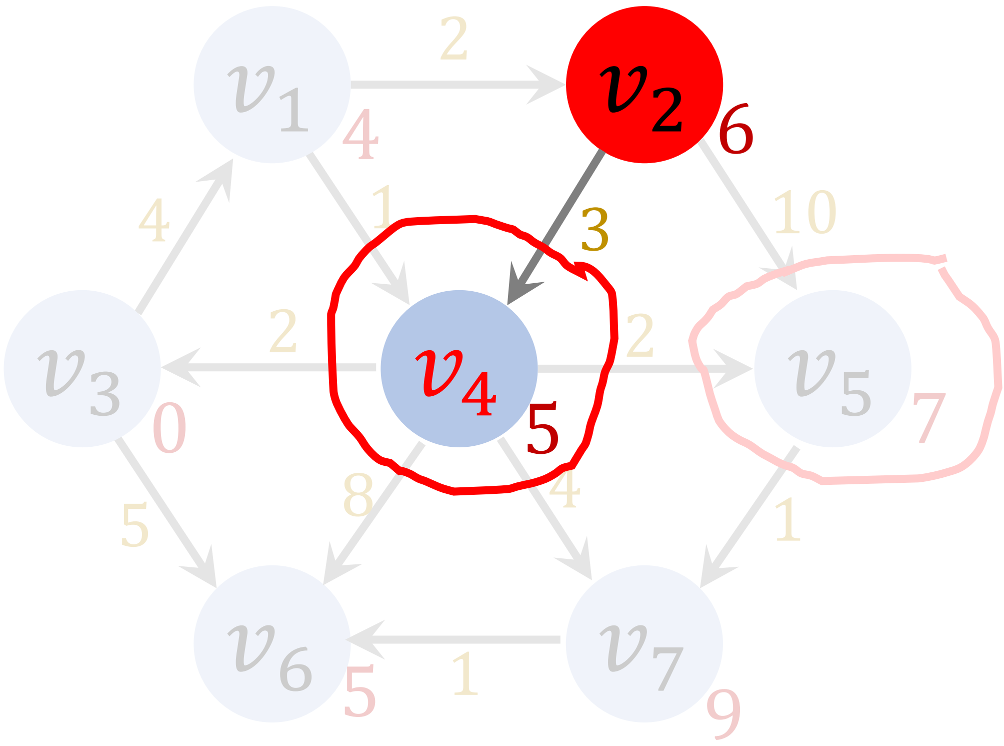


Priority Queue:

vertex	dist
v_5	7
v_7	9

vertex	dist	path
v_1	4	v_3
v_2	6	v_1
v_3	0	0
v_4	5	v_1
v_5	7	v_4
v_6	5	v_3
v_7	9	v_4

Iteration 5(A)



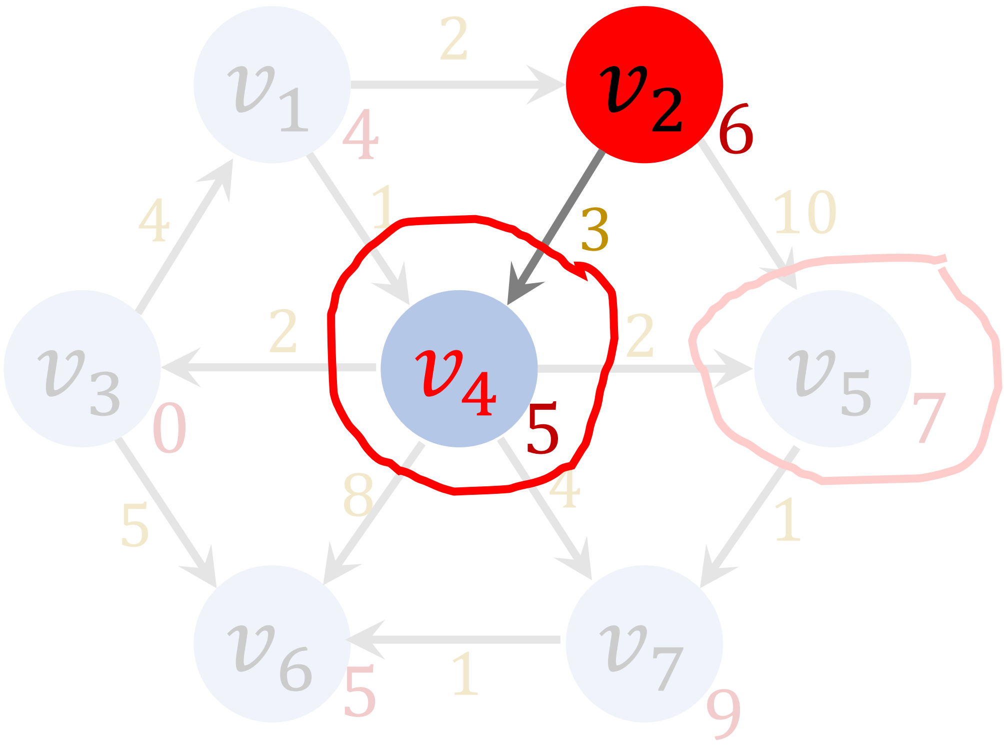
$d_{\text{new}} = 6 + 3 = 9.$

Priority Queue:

vertex	dist
v_5	7
v_7	9

vertex	dist	path
v_1	4	v_3
v_2	6	v_1
v_3	0	0
v_4	5	v_1
v_5	7	v_4
v_6	5	v_3
v_7	9	v_4

Iteration 5(A)



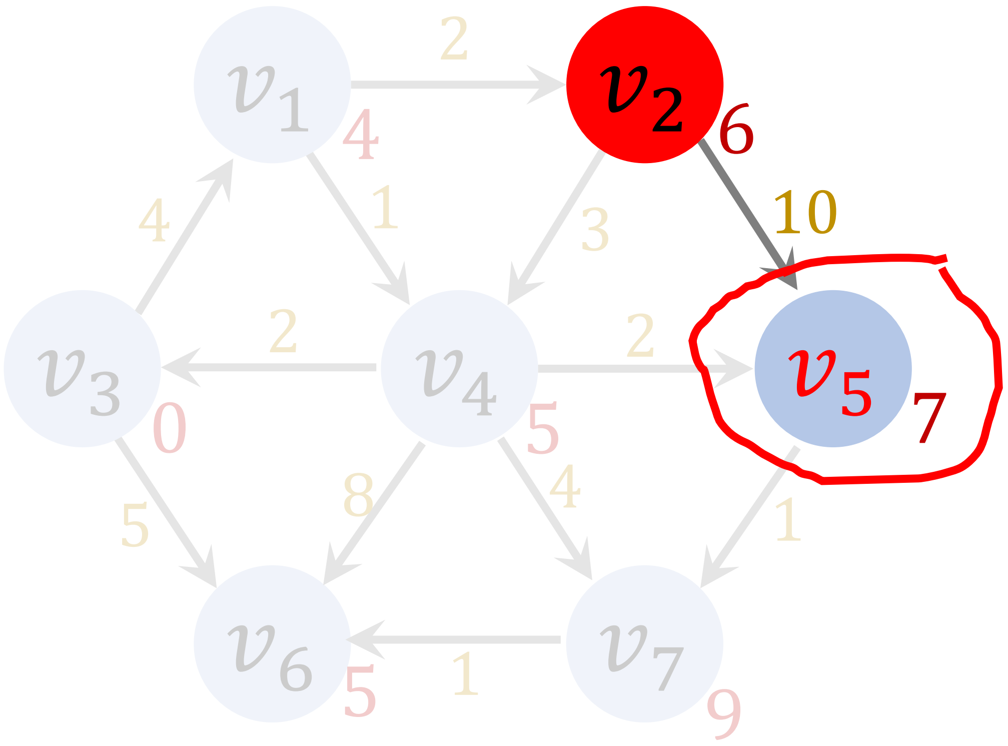
$$d_{\text{new}} = 6 + 3 = 9.$$

Priority Queue:

vertex	dist
v_5	7
v_7	9

vertex	dist	path
v_1	4	v_3
v_2	6	v_1
v_3	0	0
v_4	5	v_1
v_5	7	v_4
v_6	5	v_3
v_7	9	v_4

Iteration 5(B)

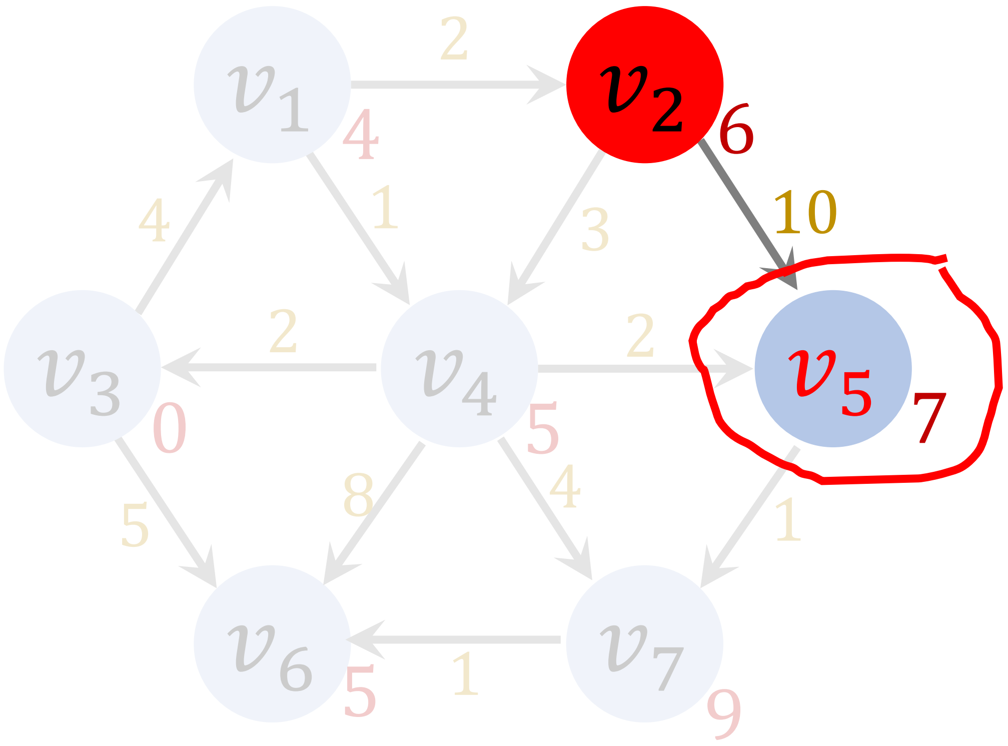


Priority Queue:

vertex	dist
v_5	7
v_7	9

vertex	dist	path
v_1	4	v_3
v_2	6	v_1
v_3	0	0
v_4	5	v_1
v_5	7	v_4
v_6	5	v_3
v_7	9	v_4

Iteration 5(B)



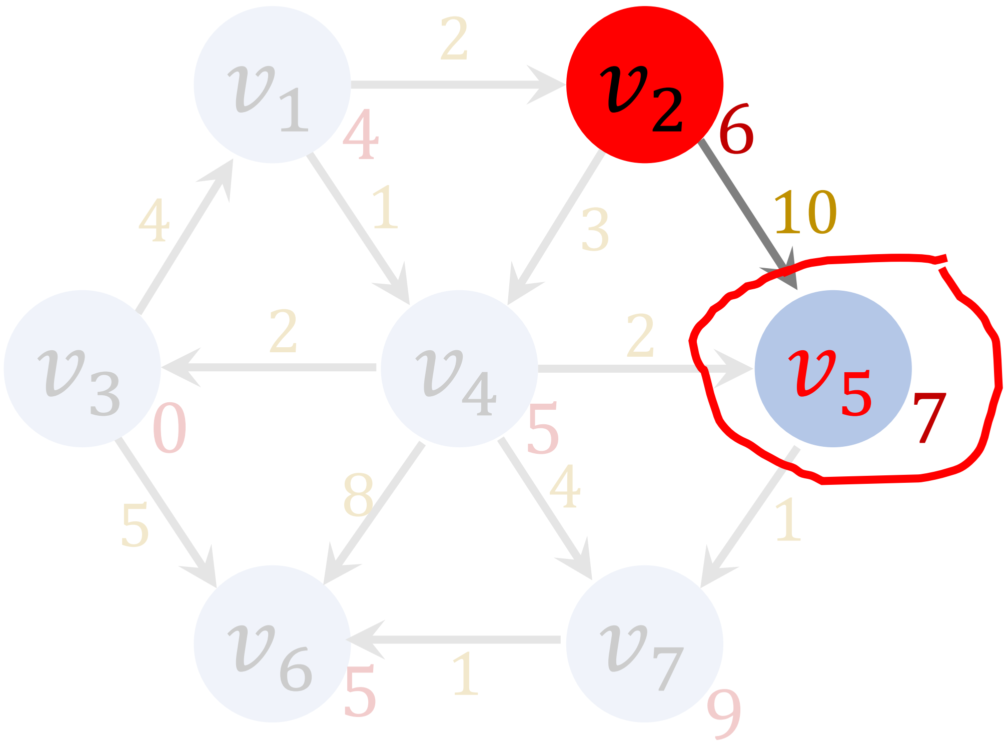
$$d_{new} = 6 + 10 = 16.$$

Priority Queue:

vertex	dist
v_5	7
v_7	9

vertex	dist	path
v_1	4	v_3
v_2	6	v_1
v_3	0	0
v_4	5	v_1
v_5	7	v_4
v_6	5	v_3
v_7	9	v_4

Iteration 5(B)



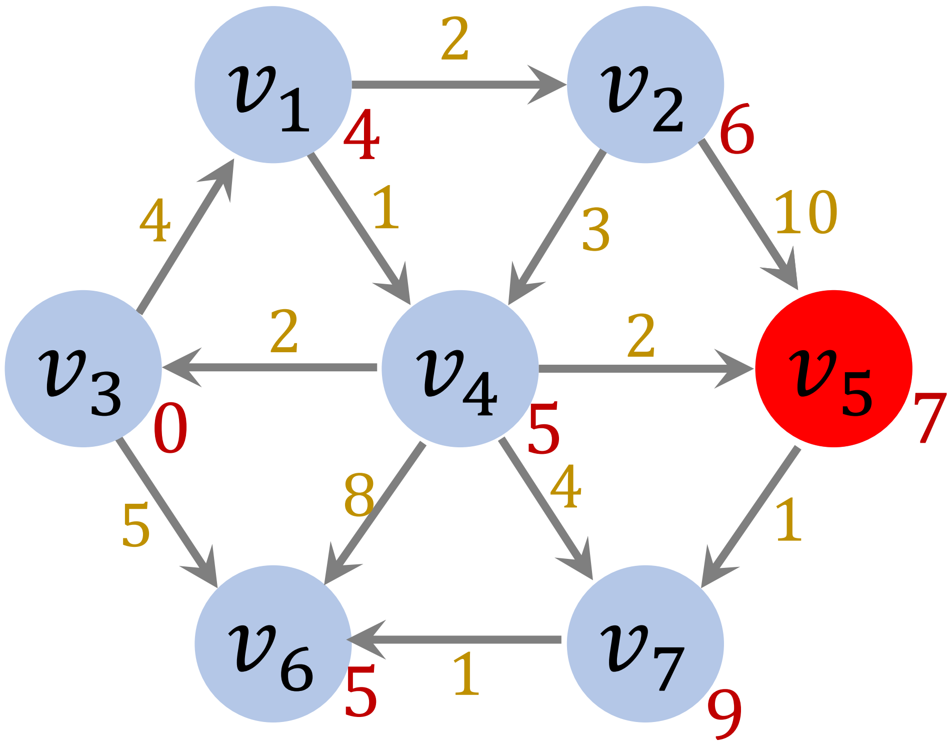
$$d_{\text{new}} = 6 + 10 = 16.$$

Priority Queue:

vertex	dist
v_5	**7**
v_7	**9**

vertex	dist	path
v_1	**4**	v_3
v_2	**6**	v_1
v_3	**0**	0
v_4	**5**	v_1
v_5	**7**	v_4
v_6	**5**	v_3
v_7	**9**	v_4

Iteration 6

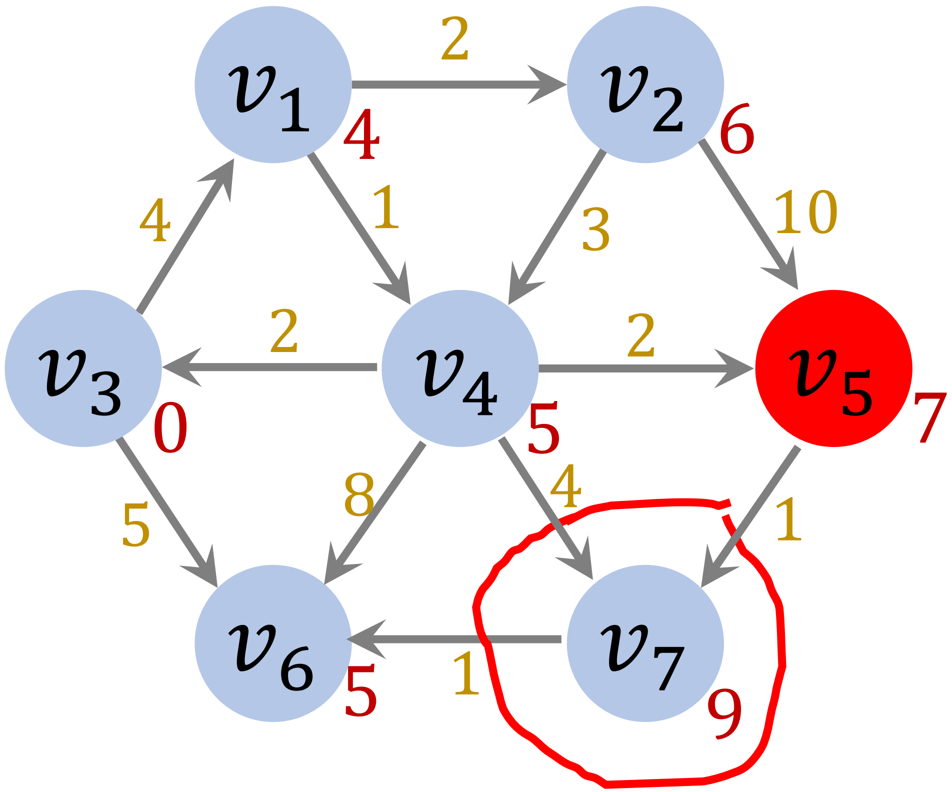


Priority Queue:

vertex	dist
v_5	7
v_7	9

vertex	dist	path
v_1	**4**	v_3
v_2	**6**	v_1
v_3	**0**	0
v_4	**5**	v_1
v_5	**7**	v_4
v_6	**5**	v_3
v_7	**9**	v_4

Iteration 6

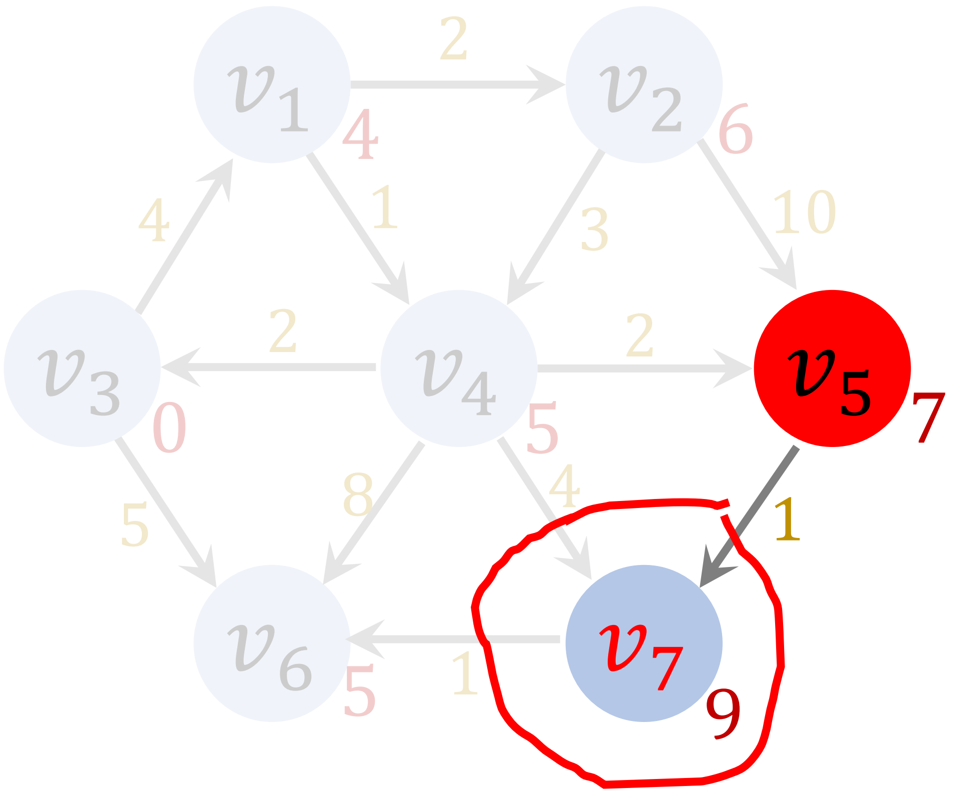


Priority Queue:

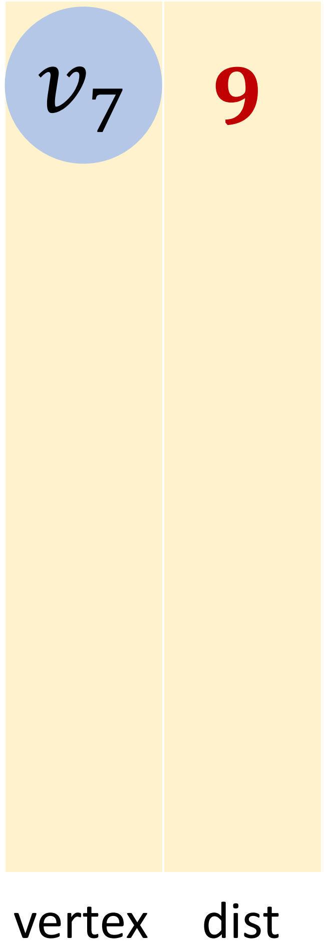
vertex	dist
v_7	**9**

vertex	dist	path
v_1	**4**	v_3
v_2	**6**	v_1
v_3	**0**	0
v_4	**5**	v_1
v_5	**7**	v_4
v_6	**5**	v_3
v_7	**9**	v_4

Iteration 6(A)

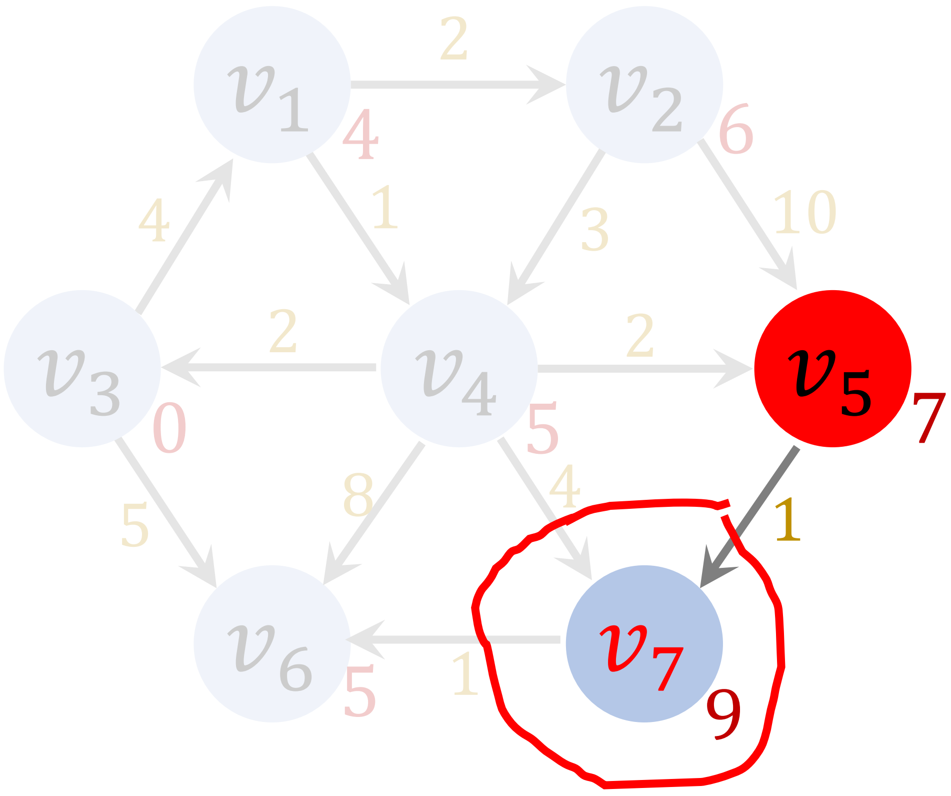


Priority Queue:



vertex	dist	path
v_1	**4**	v_3
v_2	**6**	v_1
v_3	**0**	0
v_4	**5**	v_1
v_5	**7**	v_4
v_6	**5**	v_3
v_7	**9**	v_4

Iteration 6(A)



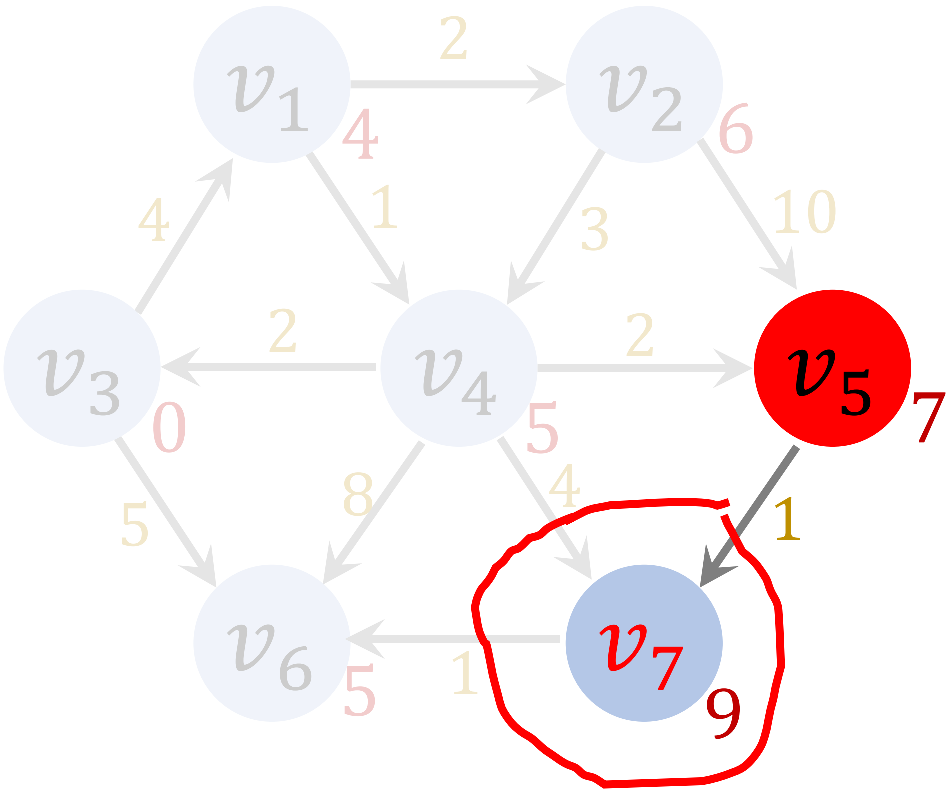
$$d_{\text{new}} = 7 + 1 = 8.$$

Priority Queue:

vertex	dist
v_7	9

vertex	dist	path
v_1	4	v_3
v_2	6	v_1
v_3	0	0
v_4	5	v_1
v_5	7	v_4
v_6	5	v_3
v_7	9	v_4

Iteration 6(A)



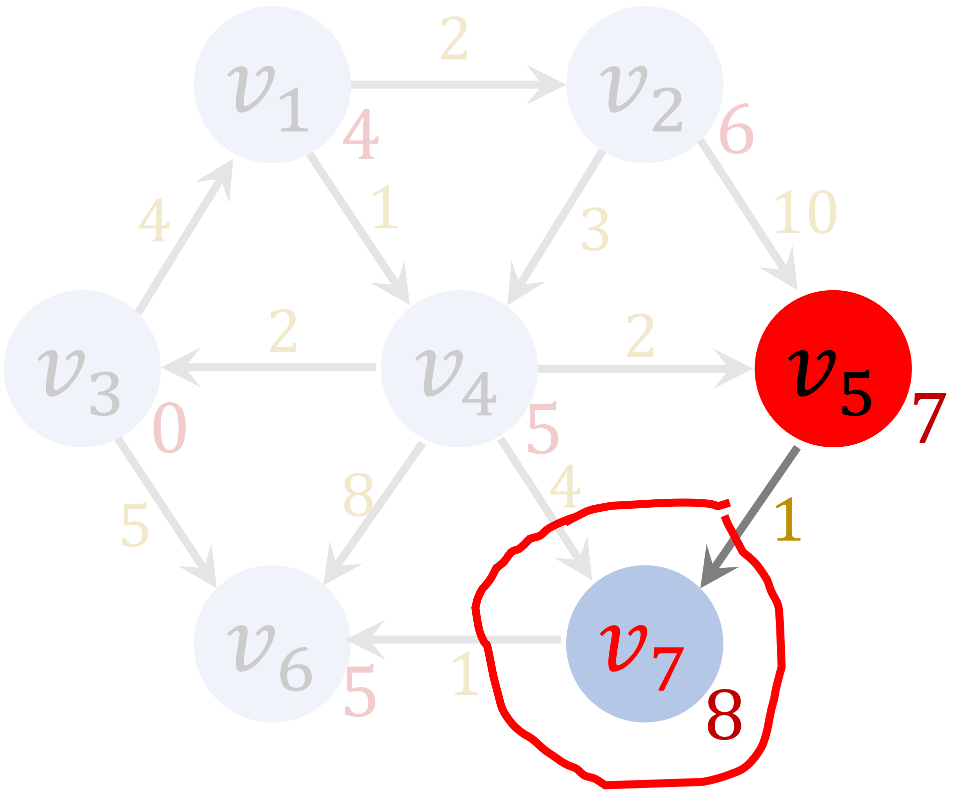
$$d_{\text{new}} = 7 + 1 = 8.$$

Priority Queue:

vertex	dist
v_7	9

vertex	dist	path
v_1	4	v_3
v_2	6	v_1
v_3	0	0
v_4	5	v_1
v_5	7	v_4
v_6	5	v_3
v_7	9	v_4

Iteration 6(A)



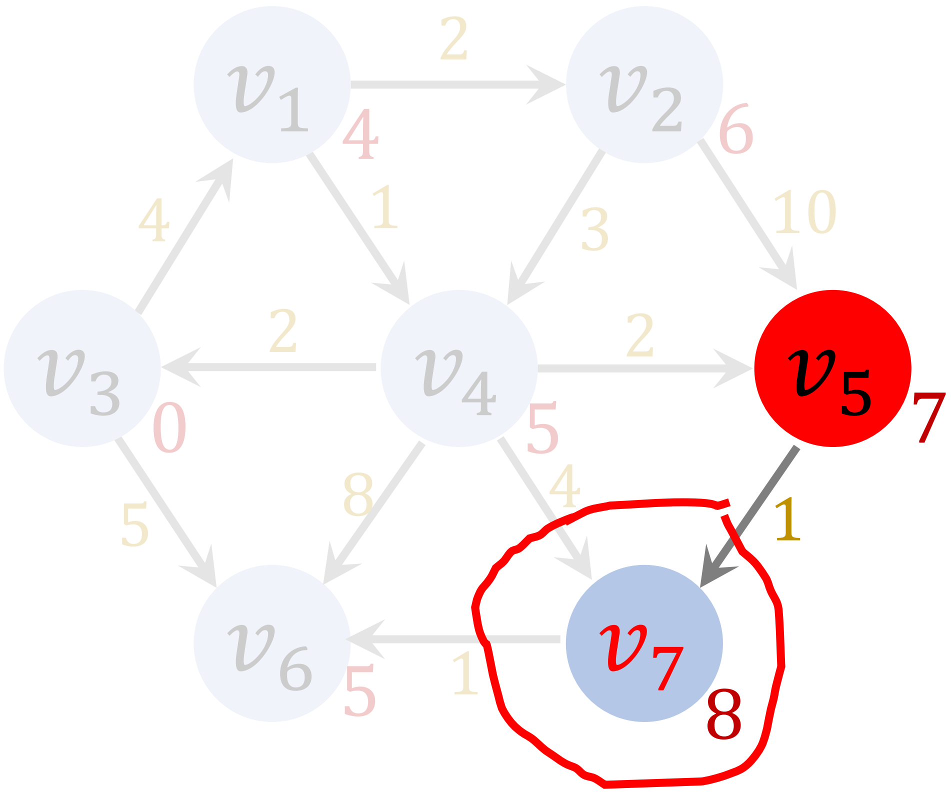
$d_{\text{new}} = 7 + 1 = 8.$

Priority Queue:

vertex	dist
v_7	9

vertex	dist	path
v_1	4	v_3
v_2	6	v_1
v_3	0	0
v_4	5	v_1
v_5	7	v_4
v_6	5	v_3
v_7	8	v_4

Iteration 6(A)

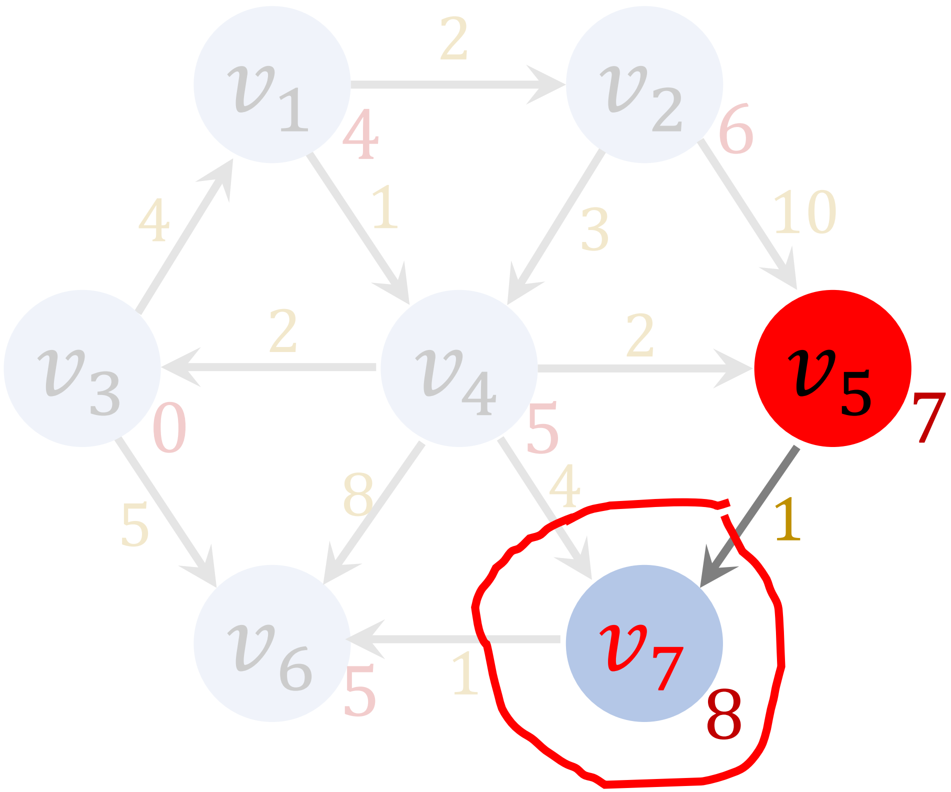


Priority Queue:

vertex	dist
v_7	9

vertex	dist	path
v_1	4	v_3
v_2	6	v_1
v_3	0	0
v_4	5	v_1
v_5	7	v_4
v_6	5	v_3
v_7	8	v_5

Iteration 6(A)

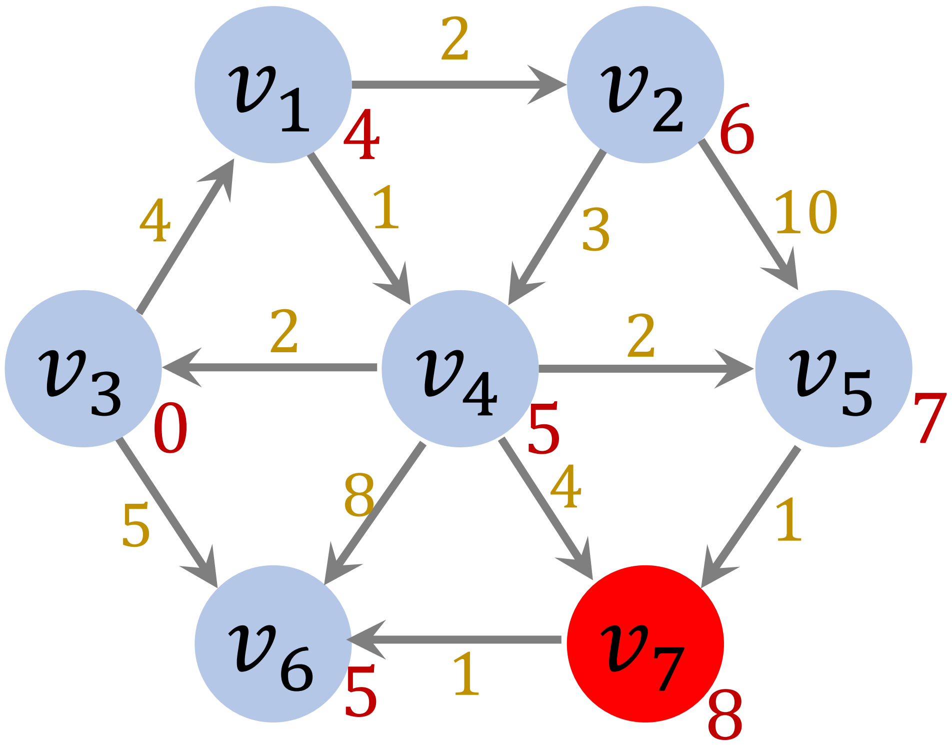


Priority Queue:

vertex	dist
v_7	9

vertex	dist	path
v_1	4	v_3
v_2	6	v_1
v_3	0	0
v_4	5	v_1
v_5	7	v_4
v_6	5	v_3
v_7	8	v_5

Iteration 7

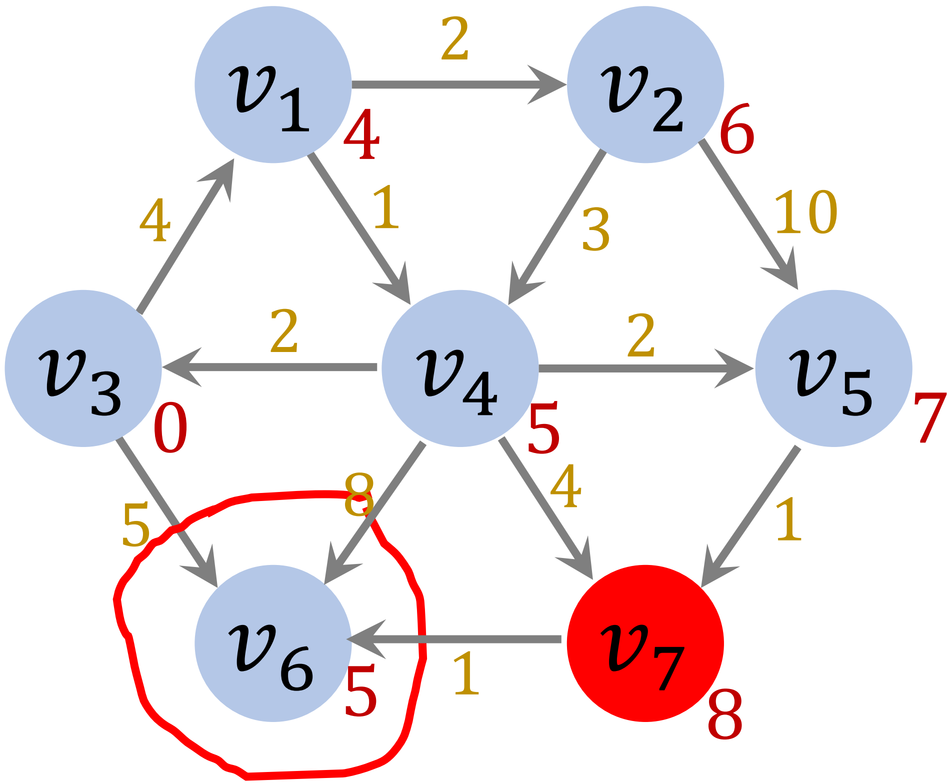


Priority Queue:

vertex	dist
v_7	8

vertex	dist	path
v_1	4	v_3
v_2	6	v_1
v_3	0	0
v_4	5	v_1
v_5	7	v_4
v_6	5	v_3
v_7	8	v_5

Iteration 7

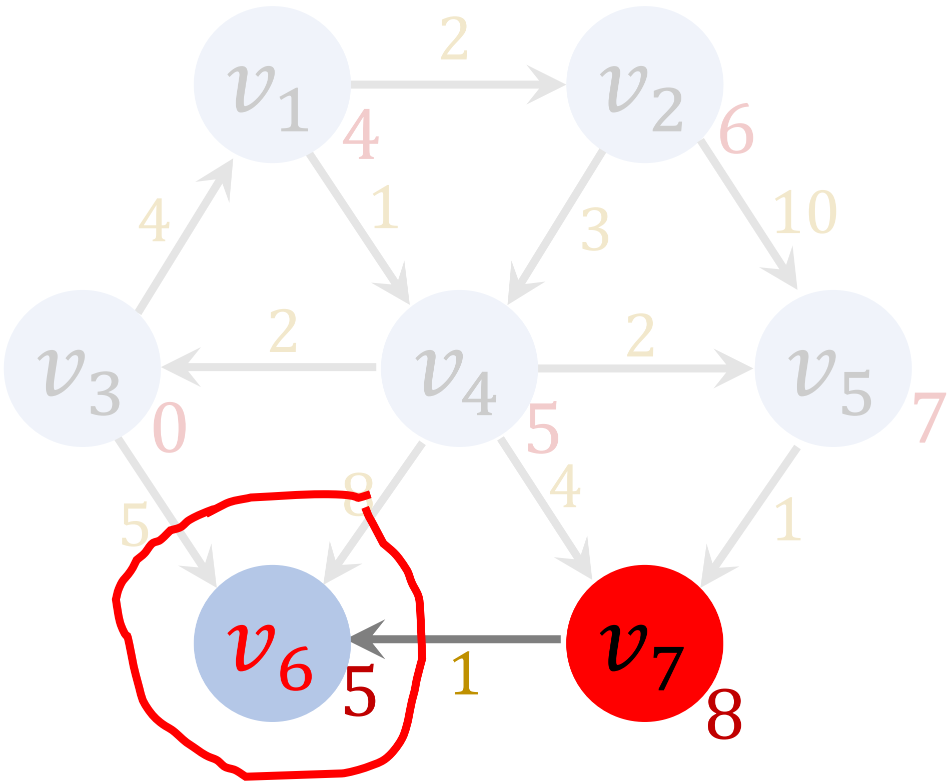


Priority Queue:

vertex	dist

vertex	dist	path
v_1	**4**	v_3
v_2	**6**	v_1
v_3	**0**	0
v_4	**5**	v_1
v_5	**7**	v_4
v_6	**5**	v_3
v_7	**8**	v_5

Iteration 7(A)

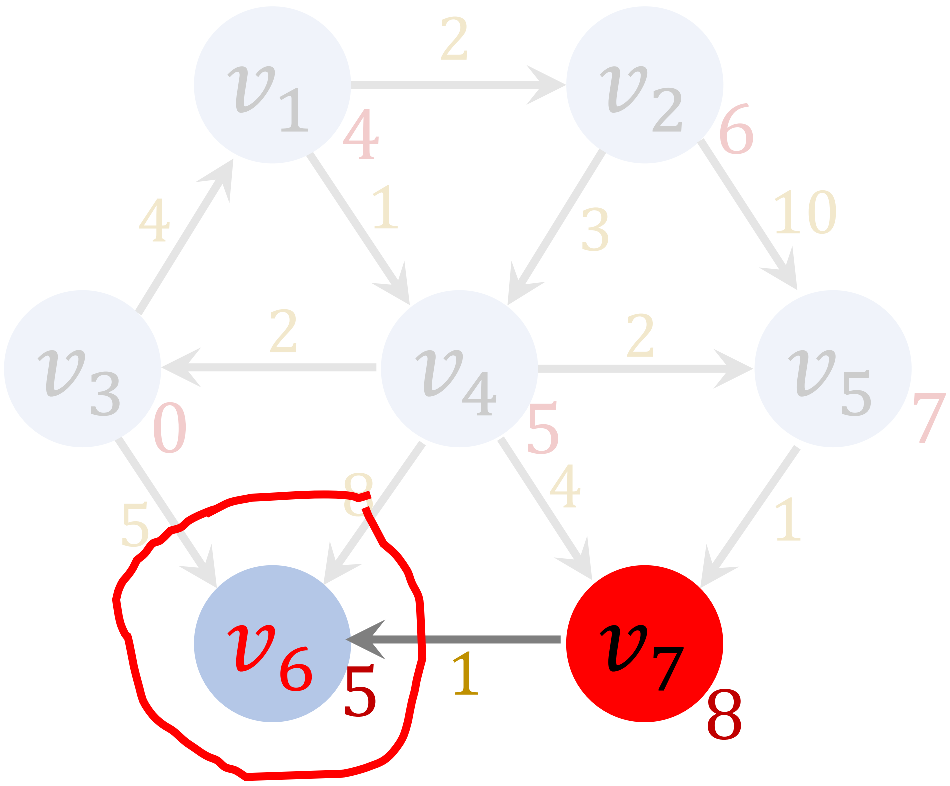


Priority Queue:

vertex	dist

vertex	dist	path
v_1	**4**	v_3
v_2	**6**	v_1
v_3	**0**	0
v_4	**5**	v_1
v_5	**7**	v_4
v_6	**5**	v_3
v_7	**8**	v_5

Iteration 7(A)



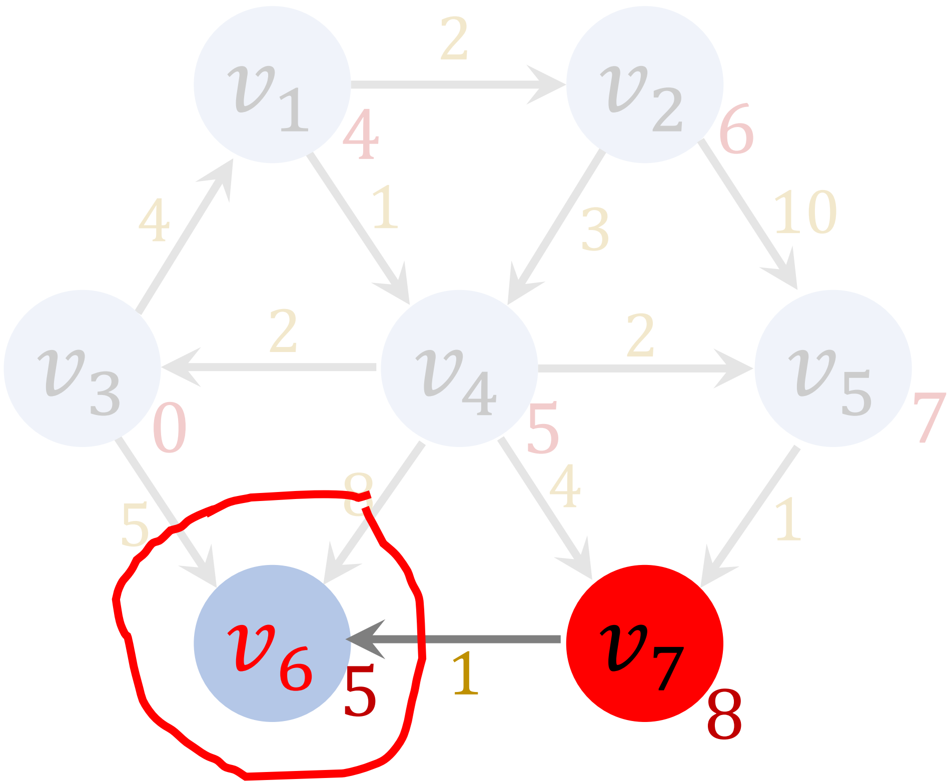
$$d_{\text{new}} = 8 + 1 = 9.$$

Priority Queue:

vertex	dist

vertex	dist	path
v_1	**4**	v_3
v_2	**6**	v_1
v_3	**0**	0
v_4	**5**	v_1
v_5	**7**	v_4
v_6	**5**	v_3
v_7	**8**	v_5

Iteration 7(A)



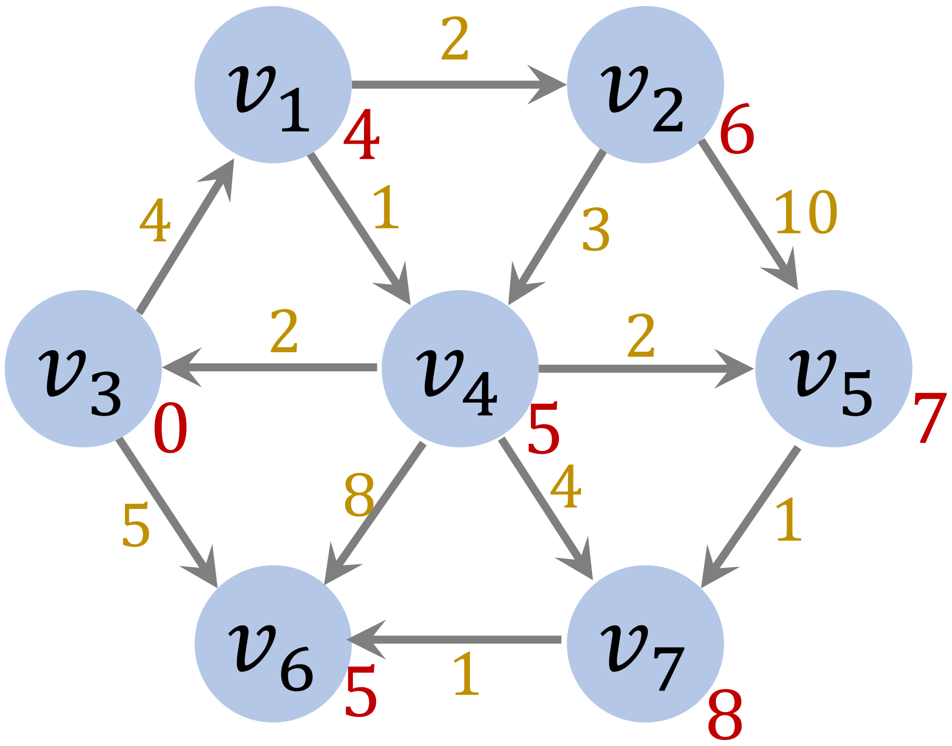
$$d_{\text{new}} = 8 + 1 = 9.$$

Priority Queue:

vertex	dist

vertex	dist	path
v_1	**4**	v_3
v_2	**6**	v_1
v_3	**0**	0
v_4	**5**	v_1
v_5	**7**	v_4
v_6	**5**	v_3
v_7	**8**	v_5

End of Procedure



Priority Queue:

vertex dist

vertex	dist	path
v_1	**4**	v_3
v_2	**6**	v_1
v_3	**0**	0
v_4	**5**	v_1
v_5	**7**	v_4
v_6	**5**	v_3
v_7	**8**	v_5

Pseudo Code

Inputs: vertices $\mathcal{V}$, edges $\mathcal{E}$, and the source vertex s.

1. Initialize an empty priority queue.

Pseudo Code

Inputs: vertices $\mathcal{V}$, edges $\mathcal{E}$, and the source vertex s.

1. Initialize an empty priority queue.
2. For each vertex $v \in \mathcal{V}$:
   a. Set $\text{dist}[v] \leftarrow \infty$.
   b. Set $\text{path}[v] \leftarrow 0$.

vertex	dist	path
v_1	∞	
v_2	∞	
$\vdots$	$\vdots$	
v_n	∞	

Pseudo Code

Inputs: vertices $\mathcal{V}$, edges $\mathcal{E}$, and the source vertex s.

1. Initialize an empty priority queue.
2. For each vertex $v \in \mathcal{V}$:
   a. Set $\text{dist}[v] \leftarrow \infty$.
   b. Set $\text{path}[v] \leftarrow 0$.

vertex	dist	path
v_1	∞	0
v_2	∞	0
$\vdots$	$\vdots$	$\vdots$
v_n	∞	0

Pseudo Code

Inputs: vertices $\mathcal{V}$, edges $\mathcal{E}$, and the source vertex s.

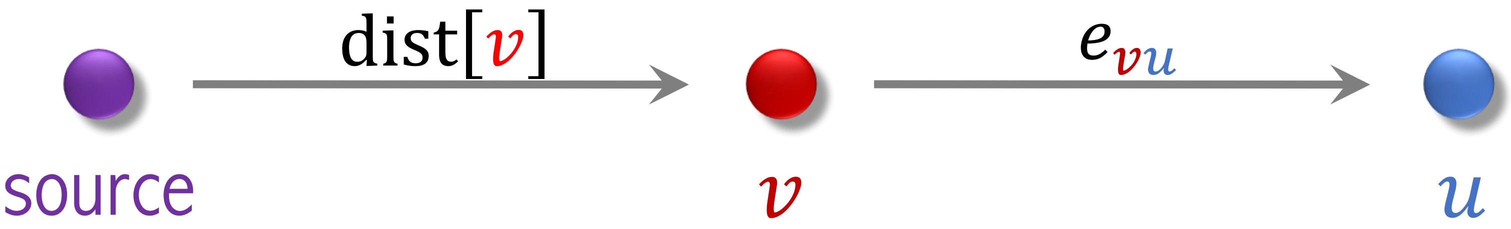
1. Initialize an empty priority queue.
2. For each vertex $v \in \mathcal{V}$:
   a. Set $\text{dist}[v] \leftarrow \infty$.
   b. Set $\text{path}[v] \leftarrow 0$.
3. Set $\text{dist}[s] \leftarrow 0$.
4. $\text{enqueue}(s, 0)$.

Pseudo Code (Cont.)

5. While the priority queue is not empty:
   a. $v \leftarrow \text{dequeue}()$.
   b. $\mathcal{S} \leftarrow \{ u \mid e_{vu} \in \mathcal{E} \}$.

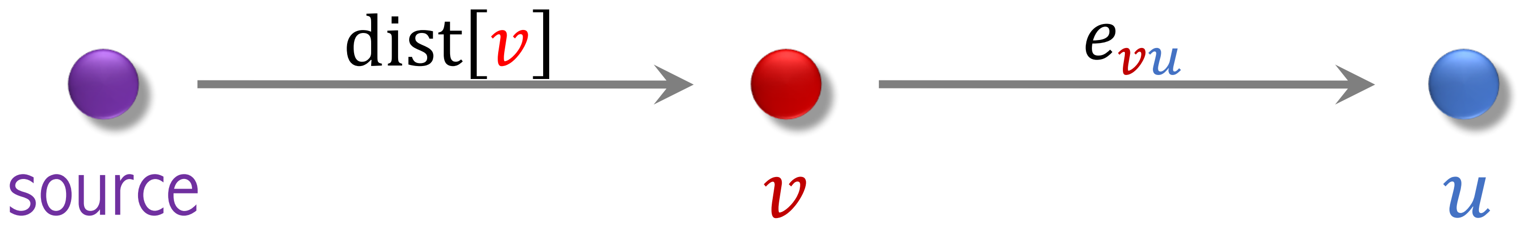
Pseudo Code (Cont.)

5. While the priority queue is not empty:
   a. $v \leftarrow \text{dequeue}()$.
   b. $\mathcal{S} \leftarrow \{ u \mid e_{vu} \in \mathcal{E} \}$.
   c. For each $u \in \mathcal{S}$:
      i. $\text{d}_{\text{new}} \leftarrow \text{dist}[v] + e_{vu}$.
      ii. If $\text{d}_{\text{new}} < \text{dist}[u]$:
         - Set $\text{dist}[u] \leftarrow \text{d}_{\text{new}}$ and $\text{path}[u] \leftarrow v$.
         - $\text{enqueue}(u, \text{d}_{\text{new}})$.



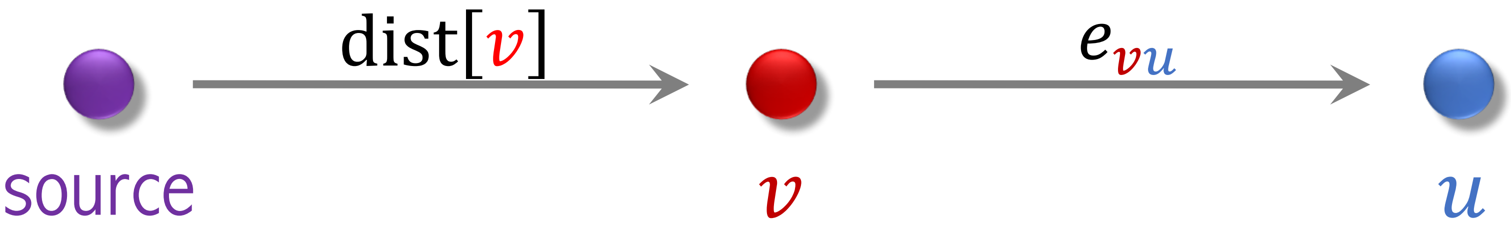
Pseudo Code (Cont.)

5. While the priority queue is not empty:
   a. $v \leftarrow \mathrm{dequeue}()$.
   b. $\mathcal{S} \leftarrow \{ u \mid e_{vu} \in \mathcal{E} \}$.
   c. For each $u \in \mathcal{S}$:
      i. $\underline{\mathrm{d_{new}}} \leftarrow \underline{\mathrm{dist}[v]} + \underline{e_{vu}}$.
      ii. If $\mathrm{d_{new}} < \mathrm{dist}[u]$:
         - Set $\mathrm{dist}[u] \leftarrow \mathrm{d_{new}}$ and $\mathrm{path}[u] \leftarrow v$.
         - $\mathrm{enqueue}(u, \mathrm{d_{new}})$.



Pseudo Code (Cont.)

5. While the priority queue is not empty:
   a. $v \leftarrow \mathrm{dequeue}()$.
   b. $\mathcal{S} \leftarrow \{ u \mid e_{vu} \in \mathcal{E} \}$.
   c. For each $u \in \mathcal{S}$:
      i. $\mathrm{d}_{\mathrm{new}} \leftarrow \mathrm{dist}[v] + e_{vu}$.
      ii. If $\mathrm{d}_{\mathrm{new}} < \mathrm{dist}[u]$:
         - Set $\mathrm{dist}[u] \leftarrow \mathrm{d}_{\mathrm{new}}$ and $\mathrm{path}[u] \leftarrow v$.
         - $\mathrm{enqueue}(u, \mathrm{d}_{\mathrm{new}})$.



Pseudo Code (Cont.)

5. While the priority queue is not empty:
   a. $v \leftarrow \mathrm{dequeue}()$.
   b. $\mathcal{S} \leftarrow \{ u \mid e_{vu} \in \mathcal{E} \}$.
   c. For each $u \in \mathcal{S}$:
      i. $\mathrm{d}_{\mathrm{new}} \leftarrow \mathrm{dist}[v] + e_{vu}$.
      ii. If $\mathrm{d}_{\mathrm{new}} < \mathrm{dist}[u]$:
         - Set $\mathrm{dist}[u] \leftarrow \mathrm{d}_{\mathrm{new}}$ and $\mathrm{path}[u] \leftarrow v$.
         - $\mathrm{enqueue}(u, \mathrm{d}_{\mathrm{new}})$.

Outputs: $\boxed{\mathrm{dist}[v]}$ and $\boxed{\mathrm{path}[v]}$ for all $v \in \mathcal{V}$.

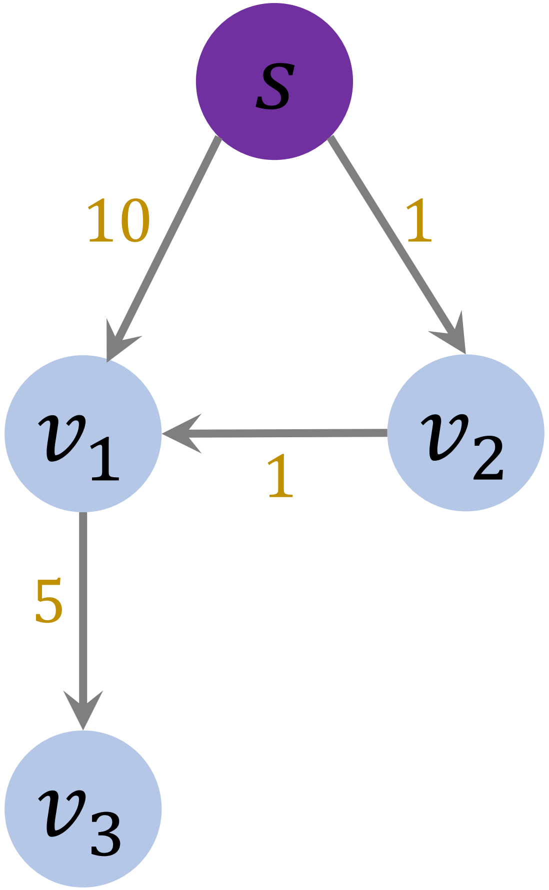
Time Complexity

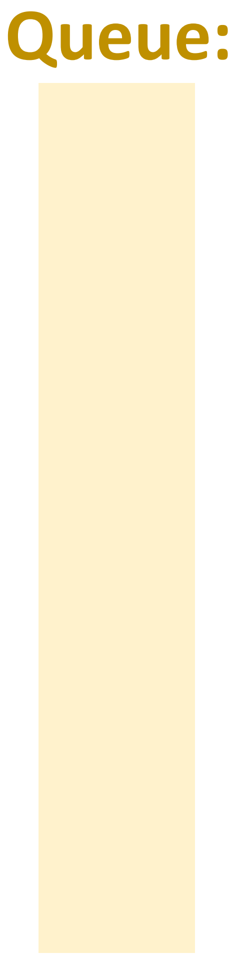
Time Complexity

- Assume all the weights are nonnegative; otherwise, Dijkstra's algorithm does not work.
- Totally $O(|\mathcal{V}| + |\mathcal{E}|)$ enqueue and dequeue operations.
- Enqueue and dequeue operations both have $O(\log|\mathcal{V}|)$ time complexity.
- Thus, the overall time complexity is $O\big((|\mathcal{V}| + |\mathcal{E}|) \cdot \log|\mathcal{V}|\big)$.

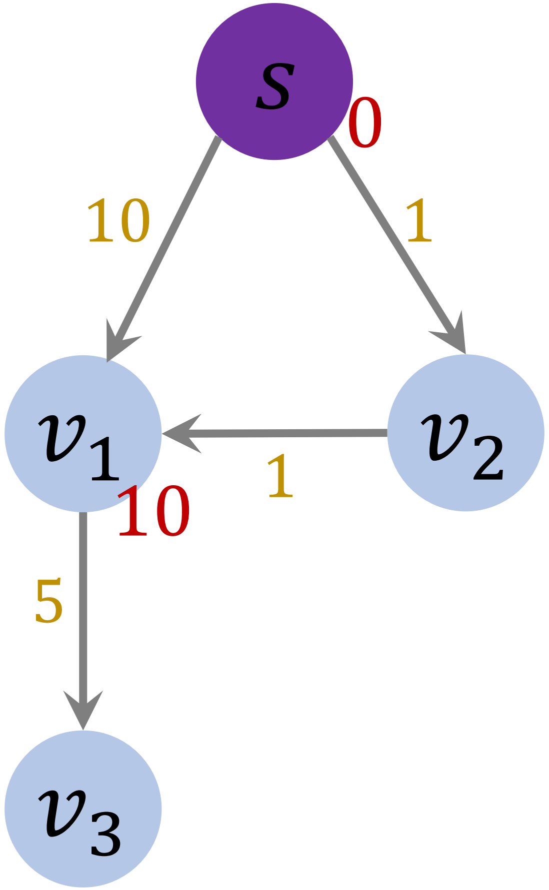
Thank You!

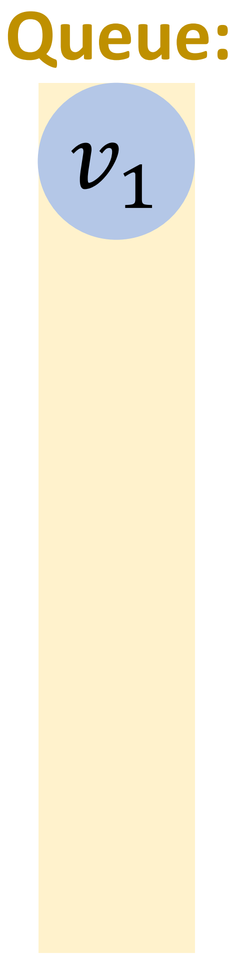
Why Priority Queue?



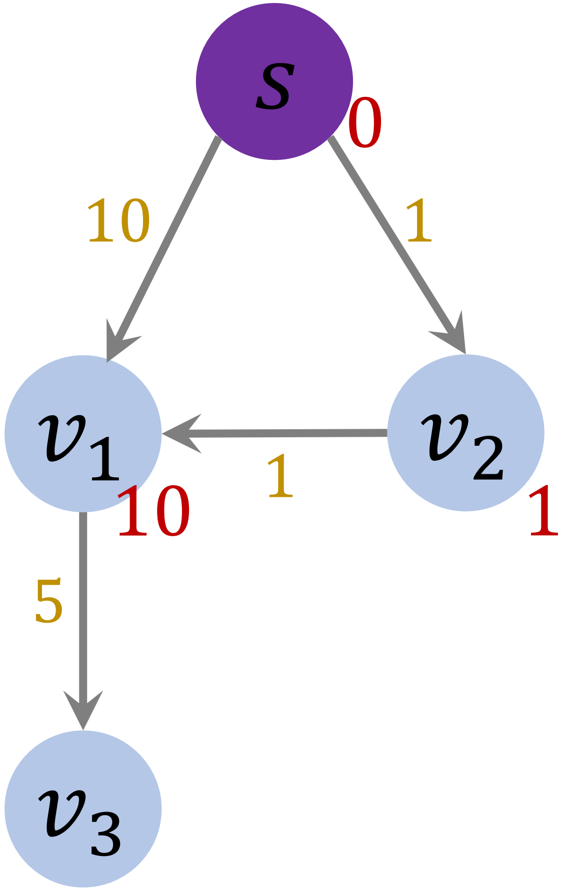


Why Priority Queue?

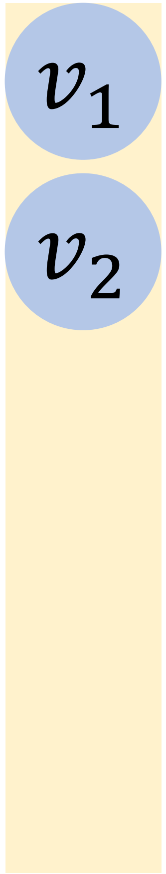




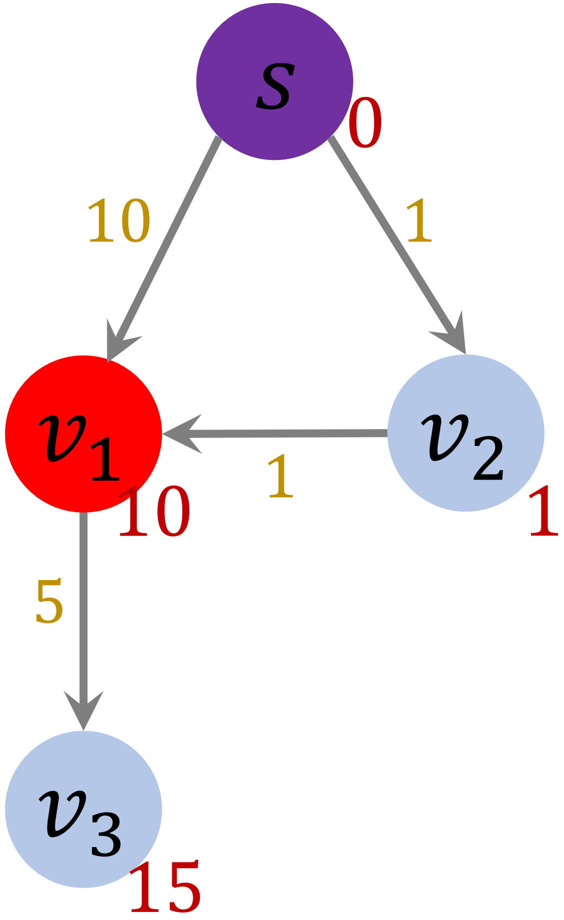
Why Priority Queue?

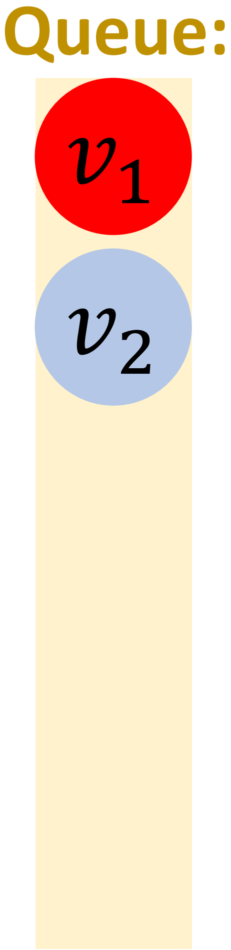


Queue:

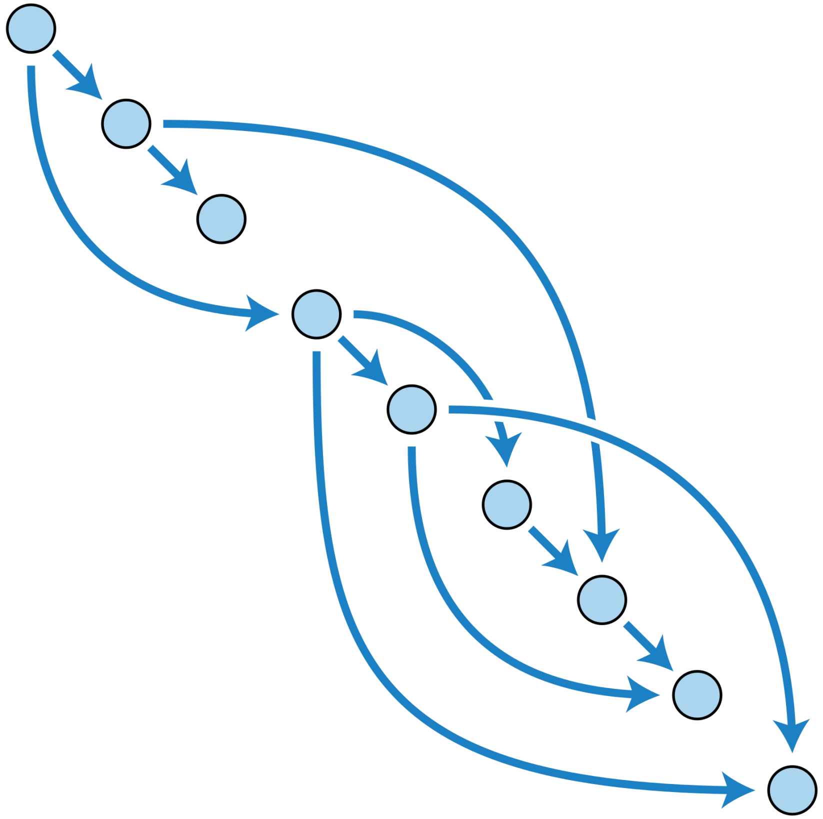


Why Priority Queue?





Directed Acyclic Graph (DAG)

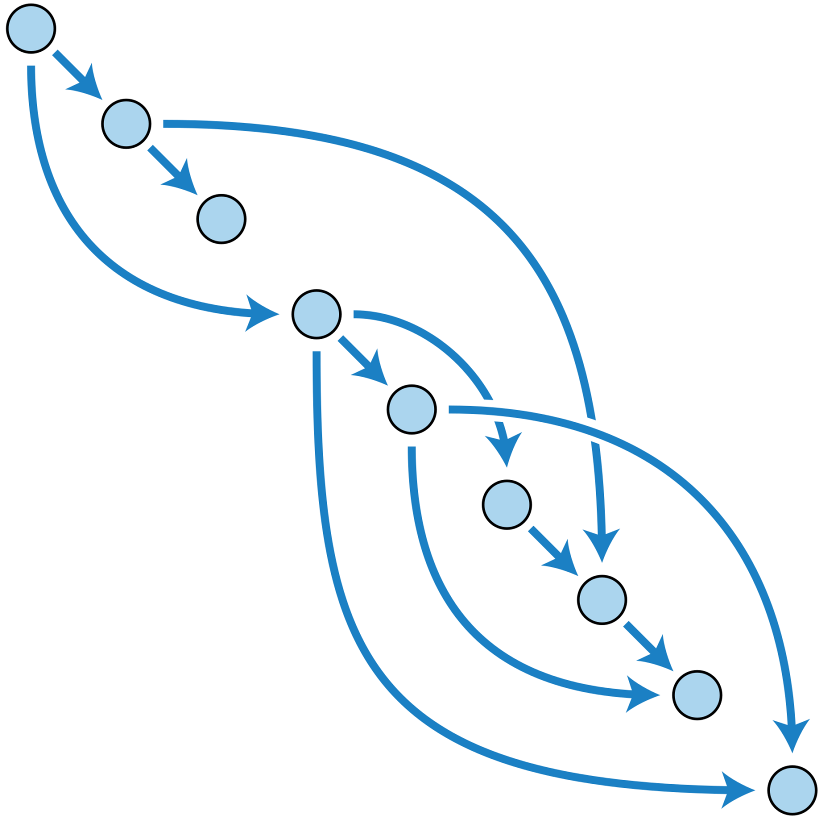


This is a DAG

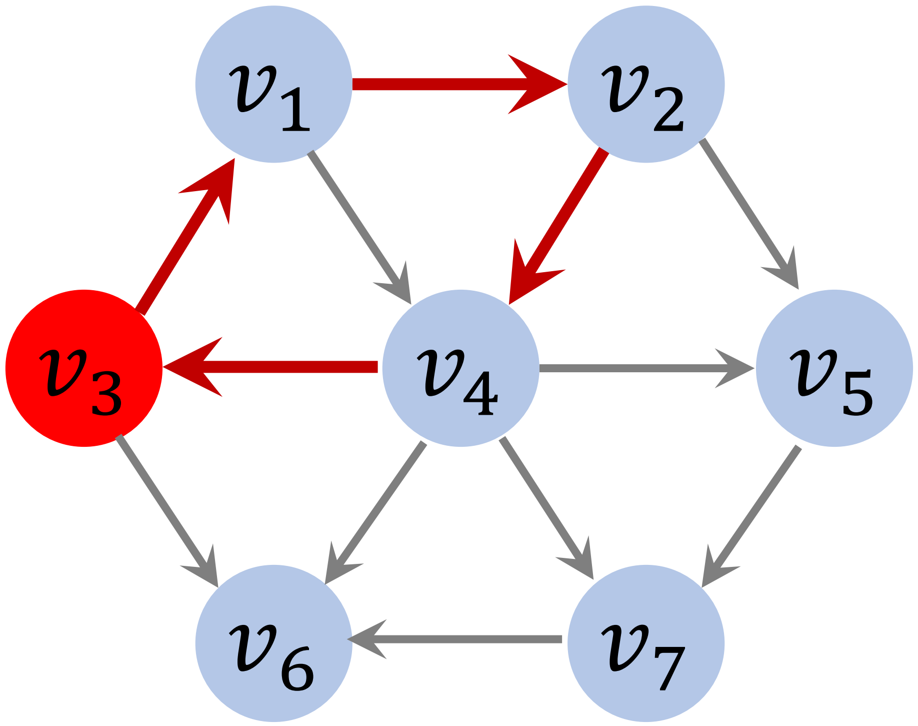
Definition of DAG

- DAG is a directed graph with no directed cycles.
- There is no way to start at any vertex v and follow a path that eventually loops back to v again.

Directed Acyclic Graph (DAG)



This is a DAG



This is not a DAG

Directed Acyclic Graph (DAG)

- If the graph is a DAG, we can use queue instead of priority queue.
  - Topological sorting is required before running Dijkstra's algorithm.
  - Topological sorting has $O(|\mathcal{V}| + |\mathcal{E}|)$ time complexity.
- Enqueue and dequeue for standard queue cost only $O(1)$ time.
- The time complexity is $O(|\mathcal{V}| + |\mathcal{E}|)$. (The same as unweighted graph.)