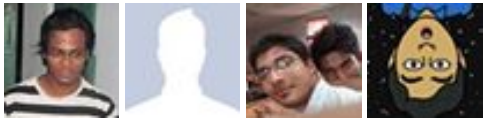
# Greedy Algorithms | Set 5 (Prim's Minimum Spanning Tree (MST))

We have discussed Kruskal's algorithm for Minimum Spanning Tree. Like Kruskal's algorithm, Prim's algorithm is also a Greedy algorithm. It starts with an empty spanning tree. The idea is to maintain two sets of vertices. The first set contains the vertices already included in the MST, the other set contains the vertices not yet included. At every step, it considers all the edges that connect the two sets, and picks the minimum weight edge from these edges. After picking the edge, it moves the other endpoint of the edge to the set containing MST.
A group of edges that connects two set of vertices in a graph is called cut in graph theory. *So, at every step of Prim's algorithm, we find a cut (of two sets, one contains the vertices already included in MST and other contains rest of the verices), pick the minimum weight edge from the cut and include this vertex to MST Set (the set that contains already included vertices).*

***How does Prim's Algorithm Work?*** The idea behind Prim's algorithm is simple, a spanning tree means all vertices must be connected. So the two disjoint subsets (discussed above) of vertices must be connected to make a *Spanning* Tree. And they must be connected with the minimum weight edge to make it a *Minimum* Spanning Tree.

***Algorithm***
**1)** Create a set *mstSet* that keeps track of vertices already included in MST.
**2)** Assign a key value to all vertices in the input graph. Initialize all key values as INFINITE. Assign key value as 0 for the first vertex so that it is picked first.
**3)** While mstSet doesn't include all vertices
....**a)** Pick a vertex *u* which is not there in *mstSet* and has minimum key value.
....**b)** Include *u* to mstSet.
....**c)** Update key value of all adjacent vertices of *u*. To update the key values, iterate through all

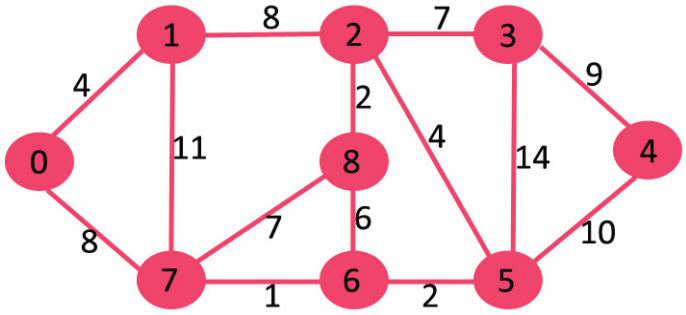




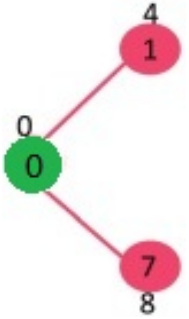
adjacent vertices. For every adjacent vertex *v*, if weight of edge *u-v* is less than the previous key value of *v*, update the key value as weight of *u-v*

The idea of using key values is to pick the minimum weight edge from cut. The key values are used only for vertices which are not yet included in MST, the key value for these vertices indicate the minimum weight edges connecting them to the set of vertices included in MST.

Let us understand with the following example:




The set *mstSet* is initially empty and keys assigned to vertices are {0, INF, INF, INF, INF, INF, INF, INF} where INF indicates infinite. Now pick the vertex with minimum key value. The vertex 0 is picked, include it in *mstSet*. So *mstSet* becomes {0}. After including to *mstSet*, update key values of adjacent vertices. Adjacent vertices of 0 are 1 and 7. The key values of 1 and 7 are updated as 4 and 8. Following subgraph shows vertices and their key values, only the vertices with finite key values are shown. The vertices included in MST are shown in green color.

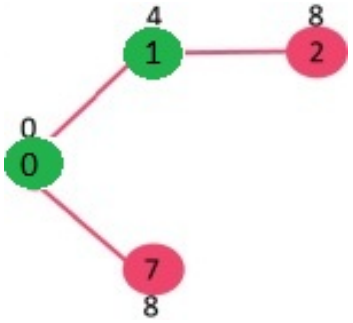



Pick the vertex with minimum key value and not already included in MST (not in mstSET). The vertex 1 is picked and added to mstSet. So mstSet now becomes {0, 1}. Update the key values of adjacent vertices of 1. The key value of vertex 2 becomes 8.

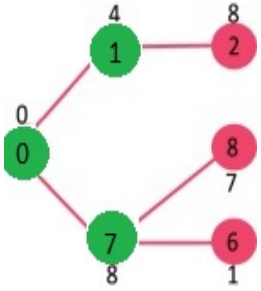
Geometric Algorithms



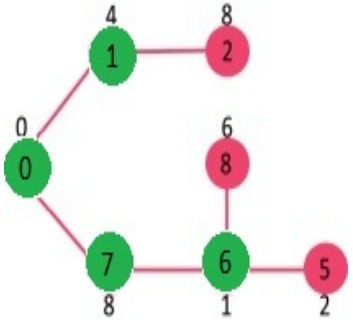
## Popular Posts



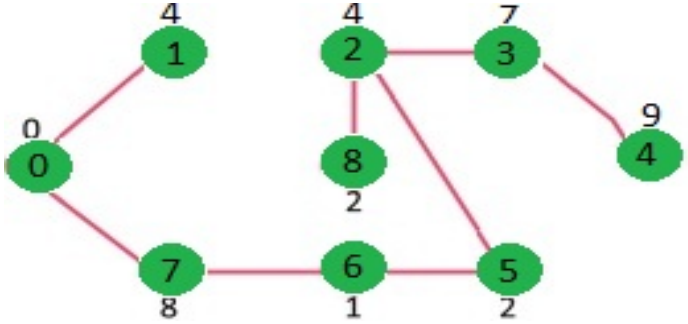

Pick the vertex with minimum key value and not already included in MST (not in mstSET). We can either pick vertex 7 or vertex 2, let vertex 7 is picked. So mstSet now becomes {0, 1, 7}. Update the key values of adjacent vertices of 7. The key value of vertex 6 and 8 becomes finite (7 and 1 respectively).




Pick the vertex with minimum key value and not already included in MST (not in mstSET). Vertex 6 is picked. So mstSet now becomes {0, 1, 7, 6}. Update the key values of adjacent vertices of 6. The key value of vertex 5 and 8 are updated.




We repeat the above steps until *mstSet* doesn't include all vertices of given graph. Finally, we get the following graph.




***How to implement the above algorithm?***

We use a boolean array mstSet[] to represent the set of vertices included in MST. If a value mstSet[v] is true, then vertex v is included in MST, otherwise not. Array key[] is used to store key values of all vertices. Another array parent[] to store indexes of parent nodes in MST. The parent array is the output array which is used to show the constructed MST.

```
// A C / C++ program for Prim's Minimum Spanning Tree (MST) algorithm.
// The program is for adjacency matrix representation of the graph

#include <stdio.h>
#include <limits.h>

// Number of vertices in the graph
#define V 5

// A utility function to find the vertex with minimum key value, from
// the set of vertices not yet included in MST
int minKey(int key[], bool mstSet[])
{
   // Initialize min value
   int min = INT_MAX, min_index;

   for (int v = 0; v < V; v++)
     if (mstSet[v] == false && key[v] < min)
         min = key[v], min_index = v;

   return min_index;
}

// A utility function to print the constructed MST stored in parent[]
int printMST(int parent[], int n, int graph[V][V])
{
   printf("Edge   Weight\n");
   for (int i = 1; i < V; i++)
```

705

Subscribe

## Recent Comments

**Aman** Hi, Why arent we checking for conditions...

Write a C program to Delete a Tree. · 33 minutes ago

kzs please provide solution for the problem...

Backtracking | Set 2 (Rat in a Maze) · 36 minutes ago

**Sanjay Agarwal** bool tree::Root_to_leaf_path_given_sum(tree...

Root to leaf path sum equal to a given number · 1 hour ago

**GOPI GOPINATH** @admin Highlight this sentence "We can easily...

Count trailing zeroes in factorial of a number · 1 hour ago

**newCoder3006** If the array contains negative numbers also. We...

Find subarray with given sum · 1 hour ago

**newCoder3006** Code without using while loop. We can do it...

Find subarray with given sum · 1 hour ago

```
        printf("%d - %d    %d \n", parent[i], i, graph[i][parent[i]]);
}

// Function to construct and print MST for a graph represented using a
// matrix representation
void primMST(int graph[V][V])
{
     int parent[V]; // Array to store constructed MST
     int key[V];   // Key values used to pick minimum weight edge in c
     bool mstSet[V];  // To represent set of vertices not yet included

     // Initialize all keys as INFINITE
     for (int i = 0; i < V; i++)
        key[i] = INT_MAX, mstSet[i] = false;

     // Always include first 1st vertex in MST.
     key[0] = 0;     // Make key 0 so that this vertex is picked as fi
     parent[0] = -1; // First node is always root of MST

     // The MST will have V vertices
     for (int count = 0; count < V-1; count++)
     {
        // Pick thd minimum key vertex from the set of vertices
        // not yet included in MST
        int u = minKey(key, mstSet);

        // Add the picked vertex to the MST Set
        mstSet[u] = true;

        // Update key value and parent index of the adjacent vertices
        // the picked vertex. Consider only those vertices which are n
        // included in MST
        for (int v = 0; v < V; v++)

           // graph[u][v] is non zero only for adjacent vertices of m
           // mstSet[v] is false for vertices not yet included in MST
           // Update the key only if graph[u][v] is smaller than key[v
          if (graph[u][v] && mstSet[v] == false && graph[u][v] <  key[
             parent[v]  = u, key[v] = graph[u][v];
     }

     // print the constructed MST
     printMST(parent, V, graph);
}


// driver program to test above function
```

```
int main()
{
   /* Let us create the following graph
           2    3
       (0)--(1)--(2)
        |    / \   |
       6| 8/   \5 |7
        | /     \ |
       (3)-------(4)
             9          */
   int graph[V][V] = {{0, 2, 0, 6, 0},
                      {2, 0, 3, 8, 5},
                      {0, 3, 0, 0, 7},
                      {6, 8, 0, 0, 9},
                      {0, 5, 7, 9, 0},
                     };

    // Print the solution
    primMST(graph);

    return 0;
}
```

Output:

```
Edge   Weight
0 - 1    2
1 - 2    3
0 - 3    6
1 - 4    5
```

Time Complexity of the above program is O(V^2). If the input graph is represented using adjacency list, then the time complexity of Prim's algorithm can be reduced to O(E log V) with the help of binary heap. We will soon be discussing O(E Log V) algorithm as a separate post.

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

## Related Tpoics:

- Some interesting shortest path questions | Set 1
- Graph Coloring | Set 2 (Greedy Algorithm)
- Graph Coloring | Set 1 (Introduction and Applications)
- Johnson's algorithm for All-pairs shortest paths
- Travelling Salesman Problem | Set 2 (Approximate using MST)
- Travelling Salesman Problem | Set 1 (Naive and Dynamic Programming)
- Detect cycle in an undirected graph
- Find maximum number of edge disjoint paths between two vertices



**Writing code in comment?** Please use **ideone.com** and share the link here.

**29 Comments** GeeksforGeeks

Sort by Newest

**steve** • a month ago

hey friends i have a question here. i have a text file as shown below:

A
A-B:3
A-C:4
B-C:2
D-C:1
D-E:2

here are the instructions: the first letter(A) on top can be any character as long
and shows our starting node(vertex),from this node we should use prim alg.to
first row is the edges and their weights and should be dynamic meaning not c

**Guest** • 7 months ago

Why have you included limits.h ?

1

**GeeksforGeeks** Mod ➚ Guest • 7 months ago

For INT_MAX

6

**Guest** ➚ GeeksforGeeks • 2 months ago

You don't need to use this at all. You can use the maximum val
last element in preorder and inorder traversal. For postorder yo
the first element so still O(1).

**Asif Raza** • 7 months ago

I used a little bit different approach for implementing same algorithm.. Its all ab problem. Have look at my C++ code for Prim's Algorithm

http://in.docsity.com/en-docs/...

3 ^ | v •

**Rohit Jain** • 11 months ago

getting trouble..nt compiling!

1 ^ | v •

**racks786** • 11 months ago

```
/* Paste your code here (You may delete these lines if not writing c
```

^ | v •

**GeeksforGeeks** • a year ago

Please try saving your file as .cpp. The program may not be compiled by all C CPP compilers. If you still get errors, then please let us know error messages

^ | v •

**Asheen Richards** • a year ago

I tried compiling this, at first I got 11 errors. I added the stdbool.h header and it the program just ends without printing anything or doing anything at all actually beginning of main to test it and it didn&#039t even print that.
It seems there is something wrong with the primMST function. I&#039m trying sure what the problem is. I&#039m codeblocks on windows by the way.

^ | v •

**GeeksforGeeks** • a year ago

Could you post some of the errors that you got?

^ | v •

**Amit Arora** • a year ago

please provide an algorithm not code.

**Alien** ➚ Amit Arora • 8 months ago

Algorithm is already provided..

Algorithm
1) Create a set mstSet that keeps track of vertices already included in
2) Assign a key value to all vertices in the input graph. Initialize all key v
as 0 for the first vertex so that it is picked first.
3) While mstSet doesn't include all vertices
….a) Pick a vertex u which is not there in mstSet and has minimum ke
….b) Include u to mstSet.
….c) Update key value of all adjacent vertices of u. To update the key v
vertices. For every adjacent vertex v, if weight of edge u-v is less than
key value as weight of u-v

1

**Syeda Shehwar** • a year ago

not getting this code:-(... it is difficult....

**Sal Alturaigi** • a year ago

I am having trouble getting this code to compile properly. I get numerous errors
command you used?

**Srinivas Giduthuri** • a year ago

Why can't I pick vertex 2 instead of vertex 7 in step 3?

**GeeksforGeeks** ➚ Srinivas Giduthuri • a year ago

You can pick 2 also. When there are edges of same wights, there may

**cooldude** ➚ GeeksforGeeks • a year ago

But if we pick 2 then mst net weight is 41 as compared to pickii take both cases nd compare

```
/* Paste your code here (You may delete these lines if
```

1

**GeeksforGeeks** ➚ cooldude • a year ago

We should same weight minimum spannning tree (trees same) by picking any of the same weight edges.
You must be missing something in your calculations.

**Murat Gurses** • a year ago

Sandeep Jain now it works perfectly fine. thanks mate.

**GeeksforGeeks** • a year ago

Yes, either put declatations first or compile it as a C++ program

**Murat Gurses** • a year ago

I don&#039t think this would compile because you should declare the for loop

int count;
for(count=0; count<10; i++); count should be outside the scope of for loop.

**Sandeep Jain** • a year ago

Could you post some of the errors here? See http://ideone.com/lVLNoP for a s

**Murat Gurses** • a year ago

is this code working correctly? when I compile it using codeblocks on mac, it g

**Murat Gurses** • a year ago

Hi. Is this code working? because when I compile using codeblocks it gives m

**Sisay Hunde** • a year ago

yes when I runn it gives one error the compiler said it, please correct it and se

**GeeksforGeeks** • a year ago

Could you please provide details of the error?

**Kunal Chopra** ➚ GeeksforGeeks • 2 months ago

I am getting error :: ) expected at line 6..
what to do?

**Momo** ➚ GeeksforGeeks • 5 months ago

For the first time " minKey" should be called after the update operation
,because it will return wrong answer for the condition "key[v] <= min" (i

the code has some error please correct it code.