minimum_spanning_tree

`minimum_spanning_tree(G, weight='weight', algorithm='kruskal', ignore_nan=False)` [source]

Returns a minimum spanning tree or forest on an undirected graph `G`.

Parameters:

G : *undirected graph*

An undirected graph. If `G` is connected, then the algorithm finds a spanning tree. Otherwise, a spanning forest is found.

weight : *str*

Data key to use for edge weights.

algorithm : *string*

The algorithm to use when finding a minimum spanning tree. Valid choices are ‘kruskal’, ‘prim’, or ‘boruvka’. The default is ‘kruskal’.

ignore_nan : *bool (default: False)*

If a NaN is found as an edge weight normally an exception is raised. If `ignore_nan is True` then that edge is ignored instead.

G : *NetworkX Graph*

A minimum spanning tree or forest.

Notes

For Borůvka’s algorithm, each edge must have a weight attribute, and each edge weight must be distinct.

For the other algorithms, if the graph edges do not have a weight attribute a default weight of 1 will be used.

Isolated nodes with self-loops are in the tree as edgeless isolated nodes.

Examples

```
>>> G = nx.cycle_graph(4)
>>> G.add_edge(0, 3, weight=2)
>>> T = nx.minimum_spanning_tree(G)
>>> sorted(T.edges(data=True))
[(0, 1, {}), (1, 2, {}), (2, 3, {})]
```